CSDN 技术改变世界 创新驱动中国

Programmer 程序员

ISSN 1672-3252

2010年11月刊

邮发代号：2-665 定价：15元


IBM软件集团 **Daniel Sabbah**
软件工程的转折点

Google公司 **Joshua Bloch**
选择编程语言就像选择酒吧


RSA信息安全有限公司 **Bill Duane**
安全正因云而改变

VMware中国研发中心 **李严冰**
未来的开放平台技术

互联网架构集结号

Sina App Engine架构分析
从#NewTwitter新界面说起
分布式文件系统FastDFS架构剖析
MySQL高可用性方案探讨 ● 消息队列漫谈
深入浅出数据仓库平台统一架构
门户网站负载均衡技术的六大新挑战

程序员 Programmer 2010年 11月刊 技术改变世界 创新驱动中国

特别感谢

（排名不分先后，以姓氏首字母为序）

Contributors

丛磊

感谢新浪 SAE 技术主管丛磊为本刊撰文，详见“封面报道”栏目。

崔海

感谢 NHN 中国技术开发中心总监崔海为本刊撰文，详见“一分钟先生”栏目。

崔金峰

感谢 58 同城网副总裁崔金峰接受本刊记者采访，详见“封面报道”栏目。

邓毅

感谢网易有道技术总监邓毅接受本刊记者采访，详见“封面报道”栏目。

Bill Duane

感谢 RSA 信息安全有限公司技术总监 Bill Duane 接受本刊记者采访，详见本期“对话 CTO”栏目。

方国伟

感谢微软软件架构资深顾问方国伟为本刊撰文，详见“技术”栏目。

冯大辉

感谢丁香园网站 CTO 冯大辉接受本刊记者采访，详见“封面报道”栏目。

高巍

感谢安卓爱普公司创始人高巍就创业中的误区这一话题为本刊撰文，详见“评论”栏目。

侯震宇

感谢百度基础平台部架构师侯震宇接受本刊记者采访，详见“封面报道”栏目。

蒋杰

感谢支付宝 BI 首席架构师蒋杰为本刊撰文，详见“封面报道”栏目。

Ivar Jacobson

感谢软件工程大师 Ivar Jacobson 与本刊分享软件开发方法的最新观点，详见“高端视点”栏目。

寇卫东

感谢 IBM 全球战略联盟新兴国家云技术总经理寇卫东为本刊撰文，针对云计算发表自己的观点，详见“高端视点”栏目。

邝宇恒

感谢腾讯广州研发中心高级工程师邝宇恒为本刊撰文，详见“封面报道”栏目。

李严冰

感谢 VMware 中国研发中心总经理李严冰为本刊撰文，介绍未来的开放平台技术，详见“高端视点”栏目。

栾义来

感谢凡客诚品架构总监栾义来接受本刊记者采访，详见“封面报道”栏目。

穆荣均

感谢美团网研发总监穆荣均为本刊撰文，详见“封面报道”栏目。

潘晓良

感谢百姓网技术总监、联合创始人潘晓良为本刊撰文，详见“架构师接龙”栏目。

潘正磊

感谢微软 Visual Studio 商业软件部总经理潘正磊接受本刊记者采访，详见“一分钟先生”栏目。

Daniel Sabbah

感谢 IBM 软件集团 Rational 总经理 Daniel Sabbah 接受本刊记者采访，详见本期“报道”栏目。

王煜全

感谢 Frost&Sulivan 首席咨询顾问王煜全为本刊撰文，评述移动互联创新心得，详见“评论”栏目。

王保平

感谢淘宝前端工程师王保平为读者介绍前端类库 KISSY，详见“产品报道”栏目。

谢旭鸿

感谢阿里巴巴用户体验设计部谢旭鸿为本刊撰文，详见“实践”栏目。

杨海朝

感谢新浪首席 DBA 杨海朝为本刊撰文，详见“封面报道”栏目。

余庆

感谢淘宝网 Java 中间件团队余庆为本刊撰文，详见“封面报道”栏目。

到底什么是云计算

本期封面策划的主题是互联网架构，这其实是我们今年继5月之后第二次聚焦行业里目前最热门的话题——云计算。可是时至今日，云计算的定义仍然莫衷一是，而且还在不断变化之中。这最明显地表现在美国国家标准技术研究所（NIST）的术语定义上，虽然已经修订到了第15版，洋洋洒洒写了两页纸，但仍然无法最终定案。

对云计算最直接的一种定义，就是基于互联网的灵活计算。它其实是互联网公司在应对不断增长的用户需求和竞争/成本压力下不得不发展出来的一系列技术的总称。在这一意义上，互联网架构实际上就是云计算最重要的技术核心。虽然云计算主要是在Google、Amazon等国外公司推动下发展起来的，但国内的互联网企业也早就有不少各具特色的云计算实践，只不过早期更多被称为动态平台、基础平台之类罢了。本期封面策划中的文章很好地证明了这一点。

可是，从更大的背景来看，云计算的意义又不限于此。云计算很可能是我们所经历过的影响最为深远的buzzword，已经有不少业内非常资深的人士在大胆预测，它很可能会成为IT发展史上最后一个大词，也就是说，成为IT革命的终结者。

那么，到底什么是云计算？

作为从业者，我们必须清醒地认识到，之前的计算方式是非常原始、落后的。无论对于普通用户还是各企事业单位，现在计算机和IT系统的使用都太难、太费劲，而且它们的性价比、效率也太低了。不是吗？想想给父母买回一台新电脑之后你还要花多少时间安装、配置、培训？想想使用各种新软件之前需要花多少时间熟悉界面、学习用法？想想要在计算机里找到某份文档、照片是多么困难？

而企业IT系统的复杂，使许多非IT机构都不得不配备人数不少的支持队伍，而且还往往疲于奔命。而相对封闭的开发方式，更使企业级开始逐渐落后于消费技术。

更惊人的是，根据美国能源部2007年的数据，从煤矿算起，经过电厂、数据中心、服务器等环节，100份能量最后只有3份用于真正有意义的计算！

是的，我们的用户花了太多时间和金钱在不产生实际效益的事情上。而在另一端，挑战也愈加严峻：更多历史的和现实事物（大到格陵兰岛崩离的冰川，小到我们的身体中各种脏器、血管、神经）的信息需要数字化，Web 2.0和SNS、物联网的发展更使每个人每件东西每天每秒都在产生更多的新数据，而人们对数据的实时性、数据之间的关联、数据所表达的意义等等方面的需求都越来越高……

云计算正是我们的应对之道。大规模互联网提供的计算能力，第一次给了我们足够的能力处理海量的数据并挖掘出其中的深意，第一次使我们可能根据每个用户的个性化需求提供真正随心所欲的用户体验，第一次使我们的用户有可能不再为IT分心，将绝大部分精力放在自己专长的事务上。在这种计算方式下，软件、硬件之间，设备之间，平台之间，社区、公司、人、物体、信息之间，过去的各种人为的或者受技术限制造成的藩篱将被打破，基础设施开放，平台开放，数据开放，包括计算机、通信、互联网、媒体内容等在内的整个信息服务产业将发生全面重组洗牌，新的生态链将形成，信息技术和能力实现了真正的平民化，大众普遍参与的新局面开始出现，群体智慧的潜力被最大程度地释放出来，更多与环境、民生、人文、艺术、生活等等相关的领域将有可能出现各种创新和创意的大爆炸。

在我看来，云计算就是计算2.0，是我们能为人类社会提供的更好的计算方式的代名词。让我们共同为之奋斗。

博客：blog.csdn.net/liujiangce

Contents

互联网架构集结号

55 封面报道 Cover Story

互联网从诞生到现在，网站的规模不断扩大，存储和处理的数据量也远远超出了人们的想象，近年来又出现了对信息实时性、多媒体需求大幅增长的现象，互联网架构面临越来越大的挑战。本期封面报道，我们特邀请了来自知名互联网企业和技术厂商的众多资深架构师，为您吹响互联网架构的集结号，全方位分析互联网架构方面的热门技术和实战经验。

Contents

2010年11月刊　总第217期

封面图片来源：维基百科 http://commons.wikimedia.org/wiki/File:Internet_map_1024.jpg

主管：中国社会科学院
主办：中国社会科学院文献信息中心
出版：《程序员》杂志社
网址：http://www.programmer.com.cn
国际刊号：ISSN 1672-3252
国内刊号：CN11-5038/G2
邮发代号：2-665
广告经营许可证号：京东工商广字0188号

总编：黄长著　Editor-in-chief：Huang Changzhu
社长/常务副总编：张悦校　President：Zhang Yuexiao
副社长：蒋 涛　Vice President：Jiang Tao
编委会：黄长著　张悦校　陈洋彬　蒋涛　曾登高　刘江
Editorial Member：Huang Changzhu　Zhang Yuexiao　Chen Yangbin　Jiang Tao
Zeng Denggao　Liu Jiang

执行主编：孟迎霞　Executive Editor-in-chief：Meng Yingxia
编辑部主任：常政　Director：Chang Zheng
责任编辑：郑柯　董世晓　高松
Editors：Zheng Ke　Dong Shixiao　Gao Song
特邀编辑：方梁　高昂　赵健平　吕娜　卢鹄翔
Contributing Editors：Fang Liang　Gao Aang　Zhao Jianping
Lv Na　Lu Dongxiang
美术设计：纪明超　Art Designer：Ji Mingchao
美术编辑：林象海　Art Editor：Lin Xianghai
流程编辑：吴志民　Coordinator：Wu Zhimin
Tel：010-64351458
Email：editor@csdn.net

发行部　Distribution Dept.010-64351431
Email：sales@csdn.net

广告总代理：北京创新乐知广告有限公司
Sole Advertising Agency：Bejing CSDN Co.,Ltd
Tel：010-64376055
Email：ad@csdn.net
Marketing Dept：010-51661202（ext 149）
Email：market@csdn.net

读者服务部
Readers service Dept.
网上订购：http://dingyue.programmer.com.cn/
读者信箱：reader@csdn.net
地址：北京市朝阳区广顺北大街33号院1号楼福码大厦B座12层
Address: B-12th Floor Fairmont Tower NO.33 Guangshun North street,
Chaoyang District,Beijing
邮政编码：100102
电话：010-64351436
传真：010-64348545

法律顾问：北京中润律师事务所 王杰
Law Consultant：Beijng Hengsheng Lawyer Firm
印刷：北京盛通印刷股份有限公司
Print: Beijing Shengtong Printing Co., Ltd.
出版日期：每月1日
Publication Date：the first day per month
零售价：RMB 15.00元　新台币 390元　HK $ 35.00（港、澳）
US $ 9.00（海外）
Retail Price：RMB 15, NT$390, HK $ 35.00,US $ 9.00

人工智能之父

——John McCarthy

■ 文 / 吕娜

他生于共产党家庭，一个关心人类可持续发展的乐观主义者。

他，一个攀岩、跳伞和驾驶滑翔机爱好者，征服过不少世界高峰。

他，首次提出“人工智能”概念，是一位被称为“人工智能之父”计算机科学家。

他，John McCarthy（约翰•麦卡锡），1971年图灵奖获得者。

从政治青年到学术人生

麦卡锡的人生像一副波澜壮阔的画卷，政治和学术是两抹最有特点的亮色。他1927年生于波士顿的一个共产党家庭，父亲做过工会组织者，搞过一些发明，母亲当过记者，热心于女权运动。因为家庭的关系，麦卡锡的童年在不断搬迁中度过，也养成了乐于阅读和思考的好习惯。

青少年时的麦卡锡聪慧过人，初中时他根据一份加州理工大学的课程目录自学完大学低年级微积分课程，也因此在1944年上大学时可以免修头两年大学数学。尽管后来二战进行得如火如荼，麦卡锡在军队任职了一段时间，但并没有影响学业，仍于1948年按时毕业，然后去普林斯顿大学研究生院继续深造。

冯•诺依曼报告引发的好奇心

麦卡锡的学术人生如何步入人工智能领域，还要从1948年9月的一次会议说起。当时普林斯顿大学主办了“行为的大脑机制西克森研讨会”，计算机大师冯•诺依曼在会议上发布了一篇关于自复制自动机的论文。这次报告激发了当时还是普林斯顿数学博士生麦卡锡的研究兴趣，他敏锐地将机器智能与人的智能联系起来，打算从事更深入的研究。

第二年，麦卡锡幸运地与冯•诺依曼一起工作，在大师的鼓励和支持下，麦卡锡决定从在机器上模拟人的智能入手，主要研究方向定为计算机下棋。此后，为了减少计算机需要考虑的棋步，麦卡锡发明了著名的α-β搜索法，这一关键问题的解决有效减少了计算量，使其至今仍是解决人工智能问题中一种常用的高效方法。

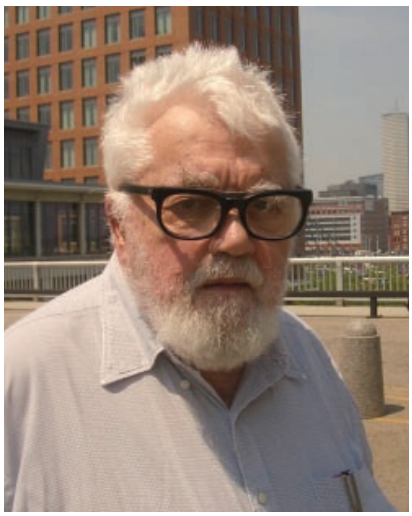
一场会议中诞生的人工智能

1952年，麦卡锡认识了贝尔实验室的香农（信息论创始人），在人工智能方面的若干深入探讨之后，他们萌生召开一次研讨会的共识。在洛克菲勒基金会的一笔微薄的赞助下，他们邀请到当时哈佛大学的明斯基和IBM工程师罗彻斯特等几位学者，参加这次具有里程碑达意义的达特茅斯会议。

达特茅斯会议历时两个多月，首次提出“人工智能”这一术语，并确立了可行的目标和方法，这使得人工智能成为电脑科学一个独立的重要分支，获得了科学界的承认。

Lisp语言和分时概念创始人

1958年麦卡锡到麻省理工学院任职，与明斯基组建了世界上第一个人工智能实验室。同年，麦卡锡发明了Lisp语言，这是人工智能界第一个最广泛流行的语言，至今仍有着广泛应用。Lisp语言与后来由1973年实现的逻辑式语言PROLOG并称为人工智能的两大语言，对人工智能的发展起了十分深远的影响。



麦卡锡另一个卓越贡献是1960年左右第一次提出将计算机批处理方式改造成分时方式，这使得计算机能同时允许数十甚至上百用户使用，极大地推动接下来的人工智能研究。他的研究成果最终实现了世界上最早的分时系统——基于IBM7094的CTSS和其后的MULTICS。

1962年麦卡锡离开麻省理工学院重返斯坦福，在那里组建了第二个人工智能实验室，并参加了一个基于DEC PDP—1的分时系统的开发。麦卡锡后来提出“情景演算”理论，吸收了有穷自动机状态转移的概念，在人工智能研究中也具有重要意义。

如今人工智能已经从实验室走进日常生活，成为一门严肃经验科学，引发了计算机使用方式的一场变革。这其中，也有着麦卡锡的一份卓越贡献。就像他在一次接受采访时所说：“我们确信技术的进步对人类有利。”

Semat计划于2009年12月由软件工程三位大师（合称“Troika”）Ivar Jacobson（UML、RUP、组件和组件架构、用例等技术之父），Bertrand Meyer（Eiffel和按约定设计之父）和Richard Soley（OMG主席）正式发起，倡导以坚实的理论、已经证明的原理和最佳实践为基础，重新发现软件工程的本质。Jacobson等撰写了三篇文章详细阐述Semat思想，本刊将陆续刊载，本文是其中第二篇。

我们为什么需要软件工程理论

Ivar Jacobson
UML三友之一，模块和模块架构、用例、现代业务工程、RUP等业界主流方法和技术的创始人。Ivar Jacobson International董事会主席。

Ian Spence
Ivar Jacobson International首席科学家，统一过程方面的敏捷应用专家。

几个星期前，Bertrand和我发表了一篇简短的文章，简单介绍了我们认为软件方法需要理论指导的原因。这篇文章中，Ian Spence和我将在上一篇文章的基础上扩展开来，更详细地讨论为什么这样一个基础理论的建立能够令我们受益。

我们最大的挑战：理解如何构造优秀的软件

我们真的知道如何开发优秀的软件吗？对大部分人来说，很显然是这样的。但是我们是否知道如何交流，以及不断改进我们开发软件的方式？我们真的了解交流和分享知识的最佳方式么？就我们在之前文章中的所见而言，显然没有！

我们站在流沙上还是巨人的肩膀上？

你是否曾经花时间研究新的方法或实践，最后却发现它只是你已经见过无数次的某种思想的改头换面？

你是否曾经烦恼过，每个软件开发新思路似乎都以过去的一切为代价，都与过去的一切水火不容？

在你看来，追逐最新的软件开发趋势是否已经变得比生产优秀的软件更重要？

你是否注意到，急着要取得进展的人们似乎丢弃了好的部分而留下坏的？他们没有从自己的经验中学习，在自己优秀的工作上更进一步，而选择将一切弃之不顾，重新开始他们认定的新事物。他们好像没有什么牢固的知识好依靠。

这种行为可以从很多地方看出来，很多团队草率地丢弃昂贵的过程和工具的投资，甚至在尝试它们之前。每个项目都采用新方法。每次工作发生变化，在手头真正的工作取得进展前，他们必须学习新方法。这是没有效率的，人们不能从经验中学习，因为他们永远从头开始。底线是，没有什么新事物能够被适当地固定下来——即使经过几种“现代”软件开发趋势，最流行的软件开发方法仍然是规范型的瀑布开发或自由hacking。作为一个行业，我们没有什么真正可以坚守的东西，而且一切似乎没有什么变化。

但现在我们能肯定敏捷会解决所有问题吗？

最新横扫行业的趋势是“敏捷”。现在，我们可以很明确地说，“敏捷”运动对软件产业做出了**非常积极的**[1]贡献。它提醒我们，软件开发中，人是第一位的，也是最重要的。事实上，这不是什么新观念，但这是重要的，而且这一

点似乎被以前更加技术导向的趋势所忽视，比如说面向对象和Java编程。通过展现一系列优点，敏捷宣言创造了某种强健和适应力强的东西，可以抵挡下一次趋势带来的变革风浪。[2]许多声称支持敏捷哲学的敏捷方法，却没能做到这一点，这是非常让人遗憾的。对一项将人的价值放在过程和工具之上的运动来说，这确实带给了我们很多“新”的过程和工具。其中的大部分已经显示出效率，通过将团队带回到之前完成的开发软件工作。但在重新聚焦到这上面之前，许多人已经迷失或迷茫，因为将新术语引入旧事物后，让人觉得这一切似乎是全新的。这个对旧思想的不断重新包装和品牌重树让软件开发团队的工作方式剧烈摇摆。对他们的工作和产品任意命名，而不是让人们远离浪费时间的工作，将精力重新聚焦在对高质量软件的开发上。

即使有些方法能够像敏捷哲学一样正确、有益，但相关的信息可能会在摇摆和炒作中丢失。我们已经开始看到对敏捷的反弹，我们担心的是利益将会丢失，当早期使用者投入下一个趋势，而晚期大众则重新主张自己的权利，拒绝采纳这些显然不再流行的东西。

有可能会发生的事情是，我们增加更多时髦的词汇和相互冲突的名词，最终为这一切喧嚣所累！

应对挑战：发展基本理论，描述软件工程究竟是什么

很显然，我们需要停止对流行和永远令人失望的简单答案的追逐，同时不能阻碍创新和新想法。为了做到这一点，人们需要停止对旧思想不断重新包装和品牌重树。相反，他们应侧重于帮助人们了解如何建立优秀的软件。但我们如何才能重点推动这一变化？我们认为，这个理论就在眼前——我们要做的只是抓住它。首先，我们应该从所有流行的方法、过程和实践开始，并从中提炼出软件工程的“真理”。然后，我们可以描述和捕捉一个最小集合的基本概念，以最小独立过程的形式——我们将这个本质物的最小集合称之为内核。

然后以这个内核为出发点，我们可以分析现有的过程和方法，并确定它们所包含的实践。从内核开始，我们可以找到一种描述实践的方式，使它们能够进行比较和结合。

现在所说的这种创造理论的方法本身并不是理论。这是我们已经做过的事情。通过研究一些方法，包括XP、Scrum和统一过程，我们的团队已经确定了20多个内核元素，我们总是做的事情或产生的东西。从表面上看，在这些被研究的方法和我们的工作方式中，有可能会出现很大差异；但在实质上，它们有相同的DNA。举例来说，你可以捕捉功能或用例或用户故事的条件，你可以在没有生命周期与统一过程的生命周期，甚至瀑布生命周期（就像有些人仍然在坚持的那样）的情况下使用这些条件。这些方法肯定有一个共同基础，能够以小的简单的内核要素集的形式被捕获。

现在，还不能冒失地声称，我们的内核提供了必要的理论。需要有比我们更多、更大的头脑来做到这一点。但是，我们会将它作为一项证据，证明它的能力和我们需要的理论就近在眼前。

但理论会带来什么影响呢？

它不仅会影响方法论、流程爱好者和学者，而且将会有利于软件开发涉及的每一个人。但是，它究竟会带来什么影响呢？

软件行业从中得到什么？

许多大公司都有自己的方法或过程，也就是一系列标准方法，搭配自己对更具体业务的想法。这些过程通常要用一本厚书或网站来介绍，大量资金被投入到归档工作中。有时，人们被训练使用这些过程，有时只是被简单告知它们在哪儿。在现实中，过程常常被忽视；仅有的被实际使用的部分是，组织中形成了“口头传统”的那些。这被解释成重新发现的自然法则：人们不看过程的书籍。新的思路引入到组织中，旧过程退出流行，而有关它们的书成为摆设。

在某些大公司甚至会出现多个过程。例如，大型系统集成商可能有十个或二十个不同的过程。有时它们很相似，但相似性背后隐藏着差异。

如果贵公司采用这种实践观，你就不需要因为一些新的性感的东西正成为流行，而抛弃整个工作方式。相反，你只需要对现有的工作方式进行改进，一次改进一个实践。你甚至可以采取那些被其他公司使用的实践，而不用丢掉似乎运作良好的现有实践。作为开始，你需要将现在的工作方式看作一个实践集合。然后寻找你的痛点，然后修补目前的工作方式，通过删除没用的实践，代之以解决这些薄弱环节的实践。一旦你理解了内核和它的使用，就很容易做到这一点。在具有多种不同工作方式的大型组织，你可以使用此方法先后改进每个工作方式，而不必强迫大家使用相同的方法或过程。

这种做法将使新实践更容易被采纳，而无须改变其他实践。想象一下，几年前，你已经引入了内核，并描述你的实践。然后，你将能够轻松引入Scrum，通过用Scrum取代项目管理中现有的实践，而无须对其他实践进行任何重大修改。展望未来，Scrum将很有可能被新的实践代替，你将能

够很容易地做到这一点了。

学术和教育界从中得到什么？

如果我们的技术学院或大学教授学生软件工程基础知识，然后训练学生在一系列良好的实践中使用该基础，那将是非常棒的。教育将会更合乎逻辑，因为它着重以独特的想法，而不是特定的思想，来形成每个方法、过程或方法论。我相信学生们会喜欢的。这里也为相关研究留下了很多空间。记住Kurt Lewin的话：“没有什么比一个好的理论更实用了。”一个好的理论使得学习和开发你的知识更容易，而不会带来过分的崇拜。这将是聪明的。大多数大学教授们在学术生涯中，从来没有真正的机会来实践大规模的软件开发。但是他们仍然不得不教授软件工程，这当然是不容易或者只是依葫芦画瓢。他们只能这样做，因为这门课在课程表上，而不是因为他们确实有什么可教的。他们没有传授理论，只是一套想法或一个特定的方法。当被问及此事时，一名成功的计算机科学家、教授软件工程课程的教授说：“令人惊讶的是，学生们喜欢沐浴在我们交给他们的烂泥塘里”。我知道这么说并不严肃，但是可以肯定这位老师并不为他做的事情而感到自豪。

一个理论，将从根本上改变这种局面。学生将学习软件的基础知识。他们将得到一种语言，来沟通软件过程、实践、模式，等等。可以想象，他们将会得到一种以内核为语法的语言和描述过程构成成分的时间的语言结构。这样的语言需要是可执行的，这样实践才会变得生动。我说这些是为了表明这些实践不仅是规范，而且也可执行。当一个项目进行时，这些实践将开始运行，而且活动实例、工作产物，实例、技能角色将被真实物创造和填充。这些方面似乎能与实践模式很好地吻合，有非常有趣的语义规则需要确定和定义。向学生打开了一个全新的世界，可以帮助他们了解软件工程的基本原理。更不用说，为对实践和理论感兴趣的研究人员打开了一个全新的世界。

方法论从中得到什么？

回顾自己1987年后的职业生涯，许多人建议我写一本有关方法论的书。当时Objectory有一些新的想法，比如说用例、用例驱动的开发（这是一个测试驱动设计、合作、序列图、组件和基于组件的开发）。其余的大部分内容都没什么特别的。实施、单元测试、系统测试、性能测试、配置、规划都是相当传统的。当然，我有整个生命周期的经验，但我不是所有事情的世界级专家。然而，为了写书，我不得不包含整个生命周期的内容，即使其中很多不是我的专长。随着我们寻找的新理论，没有任何必要再说明不包含创新的内容。你不需要写一本书来发布新想法，然后把软件开发团队需要做的一切都放进去，而只需要描述你的新实践或新模式，也许第二天你就能向全世界发布了。全世界的任何好点子都可以贡献出来并获得成功。

软件开发团队从中得到什么？

终于，软件团队将能够摆脱亦步亦趋地追随潮流所造成的无休止的摇摆，成为严格意义上的软件工程团队。团队在坚实的基础上通过优秀的软件开发实践建设和扩展知识。这个基础不会频繁变化，不会强迫你一遍又一遍学习同样的事情。它可以让你通过自己的总结，而不是出席的课程来展示专业。它可以让你轻松和无缝地引进新思路和新队友，而不会造成性能骤降或精力浪费。团队最终能够不断改进和适应他们的工作方式，迎接他们每天面对的挑战。他们将能够开发自己的知识和技能，以一种能够让他们顺利地和来自不同背景、团队和组织的其他人合作的方式，而不必一遍又一遍地重复学习同样的事情了。

最后的话

我们对软件工程的了解缺乏一个基本理论。因此，我们不断用略有不同的词再造旧方法，掩盖了真正的创新，同时让抛弃旧的不好的部分，利用新的好的部分变得困难。该理论将帮助我们大大改进软件工程教育。这将帮助我们在面对身边涌现的新想法时的反应不那么天真。最后，它也将帮助我们更快地接受新的思想。这一理论的真正受益者将是软件行业，这一点已经在许多公司得到证明。我们将能够方便地教育我们的人员，让他们加快速度；改进我们生产产品的方式；系统地重新设计（比重构程度更强）我们的产品；不断改进我们的工作方式。其结果将是更好的软件、更快的速度和大幅降低的成本。正如上面提到的，我们需要齐心协力才能做到这一点。从Scott Ambler最近的一篇文章“理论需要战略”中可以看到这种势头已经开始，但仍有许多工作要做。

我们已经证明它的有效性，但我们仍然要做许多工作才能建立一个公认的标准，必须在一组专家和权威之间建立共识才能完成这点。我们期待着与这些专家的合作。

要了解更多参与的方式，请访问 www.ivarblog.com 或者在博客“软件工程需要一个理论” 上给我们留言或者发电子邮件到a_theory@ivarjacobson.com。

- [1]如果你已经在世界各地的会议上看到我们最近的发言，你应该了解，我们是敏捷的狂热粉丝，是敏捷宣言的支持者。
- [2] 新的趋势一定要跟随敏捷趋势，就像黑夜要跟随白天一样。

云计算是中国的机会

什么是云计算？不同人有不同的定义，在我看来，云计算是为企业打造一个全新的IT消费和交付的模式，云计算将给我们带来根本性、革命性的变化。

云计算为什么是中国的机会？

在信息产业革命中，中国已经失去了PC时代和互联网时代的话语权，中国也因为没有掌握核心技术走了很多弯路。目前我们的经济发展还是靠制造业赚得的一点薄利润，没有创新、没有附加值，所以我们应该向世界先进的企业学习。在云计算时代，中国一定要抓住机会，争取获得话语权。

中国有很多制造行业，这些行业的特点是厂点多、分布散、信息化差。为了提高生产效率，他们想做IT的集中管理，但是传统的IT帮助有限。可是对云计算来说，这些特点恰恰是机会，因为云计算的最大亮点之一就是可以不分地域、规模而提供IT服务。通过云计算，企业可以得到它需要的IT服务。云计算给中国企业带来了机遇，我们不再需要像国外企业一样从头开始打造企业的IT系统了。我们可以和它们一样，利用云计算提供的IT服务满足企业的IT需求。从这个意义上说，中国企业和国外企业是站在一个起跑线上的。

过去购买软件、硬件设备，是需要计入固定资产投入的。在一般企业或者政府中，需要专门的审批程序和流程，而一旦审批的流程时间过长，对于企业或者政府的应用来说，则会浪费不少时间。云计算不需要固定资产投入，IT服务费用是公司运营费用的一部分。云计算给业务部门节省了很多在传统IT中固定资产审批需要的时间，从而能够让用户快速部署新的应用，抢占商机。

另一方面，企业的固定资产的价值不可避免地是逐年递减的。对上市公司而言，固定资产投资越少，企业的利润越高。过去购买一个设备需要一千万，而现在有了云计算，每个月只需要支付很少的钱购买所需的IT服务（包括IT设备的服务），公司的财务报表比原来的报表健康得多，对于投资者来说有很大的吸引力。此外，云计算可以帮助公司节省成本，比如说，服务器虚拟化可以使总体拥有成本节省30%~70%，存储虚拟化则能将容量需求减少多达25%，而桌面虚拟化可以帮助客户将总体拥有成本节省40%。

云的交付方式带来了IT的经济模式改变。云计算的虚拟化、标准化、自动化使得企业或者政府不再需要重新打造自己的IT系统了。企业或者政府可以根据自己目前的需求，通过云计算的标准化菜单，预订所需的IT服务。当需求增加或者减少时，企业或者政府可以相应地增加或者减少自己所需要的IT服务。这种弹性IT服务是云计算带给企业或者政府的巨大好处。

寇卫东
IBM全球战略联盟新兴国家云计算总经理

IBM将全世界分为两大市场，一是成熟市场（包括美国、加拿大、西欧、日本），二是新兴市场（包括中国在内的许多国家），目前我负责新兴市场的开拓，挑选战略合作伙伴一起推动云计算技术的发展。IBM正致力于云的交付应用工作，我们可以提供全套的云管理和资源平台，满足客户建造私有云、公共云、混合云所需的计算、存储、网络和软件，最终目的是使得企业在提出云服务需求的数小时内便为企业提供所需的“云”平台。

云计算也是程序员的机会

云计算的机会首先在于后台基础设施的整合问题，这也意味着商业智能化的重要性在逐渐加大，包括一些协同软件、平台软件也是需要开发者开发的。

其次就是安全性和可靠性问题，目前基于云计算的安全问题还没有做到让所有人足够放心。作为制造厂商，我们必须努力提高技术能力让用户在云计算的安全性上能够放心。可靠性方面，如果云计算服务厂商不能按需提供服务或者无法做到准时交付，就会让用户失去信心。再有就是网络资源问题，如果网络资源不够，云只能存在于理论上，这迫使网络提供者做革命性的变化，我们周边的国家比如韩国和日本都在大量扩充网络速度，未来基于云应用的网络传输速度应该是以G来计算的。

最后是终端应用的市场机会，包括手持终端设备如何与云计算联系起来。随着技术门槛的降低，程序员的群体会逐渐扩大，会有很多人在智能化的终端设备上开发应用程序，除了一些特殊应用外，很多人都会变身程序员进入云计算领域。

另外，云计算还要跟物联网联系起来，包括嵌入式开发的发展以及RFID和其他传感器技术，这都需要我们程序员去做。总之，程序员这个群体在云计算中可以发挥的空间非常大，也有很多机会，能否把握住，就看大家了。

未来的开放平台技术

李严冰
VMware中国研发中心总经理

开放平台技术的现状

开放平台的出现是为了将资源进行更好的整合，这种资源包括编程的模型、超大文件系统、海量的数据库、监控调度的管理等等，这些资源如果最终进行有机的整合，能够集成于一种分布式的、并行的计算平台上来提供各种资源的输出。

开放平台的优点很多：首先，技术门槛降低让应用更容易生成，从而间接鼓励更多的商业模式创新，尤其是资金花在软件和硬件的比例会减少，给初创公司带来更大的生存空间；其次，可以有更多的平台服务架构在现有的PaaS上，使得服务的种类多样化，这也会促成生态链的形成；最后，公司的合并门槛降低，如果两家公司用的是同一个平台服务，那么就没有技术整合的问题了。

但与此同时我们也面临越来越多的挑战，首先是需要面对不断移动的海量数据，整合流行的SNS服务和提供安全的支付问题都是我们面临的新问题。现代应用程序也需要易于使用，能够提供随时随地的访问以配合其他项目。企业应用将越来越需要跨平台的功能，应用开发者需要在各种SNS网站上整合服务，比如Facebook、Twitter等，以及针对包括Android和iPhone在内的移动设备。

一个现实的问题是，API和数据都还不是标准化的，使得应用迁移变得复杂。两个平台之间必须相当对称，以便应用能够在外部公共云和内部私有云之间迁移和互操作。再有，网络的连接性也是一大问题——当你的应用因为任何一端的网络而没办法连上平台服务时，你可能没有任何其他的备份方案。

可以看到，多数企业依然存在一些遗留应用，同时未来还将出现更多新的应用。新应用可能依然会采用传统的工具进行开发，但也有采用开源的、基于Java的新开发方式，以便加快应用开发的速度。

但是在哪里追踪你的源代码？在哪里跟踪你的问题？又在什么地方进行持续性的集成构建？所有这一切问题，开发人员不得不依靠他们自己的力量来解决。

从目前来看，开发一个新的桌面应用程序到云是一个脱节的过程，内部开发过程和通过网络进行的测试和部署过程之间并没有直接联系。

总而言之，我们目前还缺乏优秀的、能够统一的应用管理办法，我们需要新的弹性数据和缓存，以解决传统的关系数据库上出现的性能瓶颈，开发者也需要更简便地使用数据高速缓存平台。

VMware与开放平台

开放平台涉及的不仅是技术上的突破，更需要商务模式上的突破。开放平台需要首先提供一个基本的服务，然后通过开放自身的接口，使得第三方开发者得以通过运用、组装其接口以及其他第三方服务接口产生新的应用，并且使得该应用能够统一运行在这个平台之上。开放平台模式成功的要点在于，通过自身服务和第三方应用的互利互惠，提高用户对平台网站的粘性和使用程度，进而提高获利。同时，通过利益分摊，达到平台自身和第三方应用循环刺激而产生的滚雪球式的增长。

Java是企业应用的主要开发语言，在如何帮助Java开发人员接触云计算这一点上，VMware投入了大量的精力。我们一直在致力于为企业的Java开发人员提供创新的生产工具，让他们能够创建和部署Java应用程序在数据中心或者是私有云和公有云中。

虚拟化技术使得应用对于传统操作系统乃至基础架构的依赖性越来越低。这样一来，用户便可以采用Java，自己进行开发或委托外部公共云提供商进行应用的开发，无论这种Java开发工具是基于VMware提供的SpringSource框架还是Rabbit的框架。

Spring会提供OAuth 2.0认证让开发人员能够安全地访问各种SNS网站，并从各种SNS网站交互文件数据。一些新的移动设备的Web应用程序已经支持Spring MVC框架，它能够提供客户端的自动检测和内容的调整。

基于vFabric平台下的Code2Cloud的新服务将提供一个完整的开发环境，同时提供在私有云或者公共云中轻松部署应用程序的方案。有了Code2Cloud，一旦开发人员修复了Eclipse的某个缺陷，托管代码将即刻进行构建、测试和部署。任何在运行时的问题检测以及详细的错误报告将在跟踪器和IDE上得到立即显示。

移动领域、非关系型数据库以及各种SNS平台，都是VMware目前关注的重点。随着时间的推移，VMware将可能增加更多的语言比如，Ruby、PHP等。

IEEE Software
2010.10

表达涉众的关注

一些涉众（stakeholder）的关注（concern）因为可用架构的视角、框架，或架构描述语言（ADL）得到了良好解决，有的则没有被目前的现成方法所表达。因此，本期的主题是：探索在多重涉众的关注上进行架构的空间，以及寻找在该空间内能够帮助架构师的解决方案。

本期的文章从不同的方面呈现了一个或多个涉众关注的技术。有些是在软件架构内面临的问题；另外一些则包括了软件工程其他分支对涉众关注和架构视角的见解。

在《需求驱动的面向服务交互设计》一文中，解决了面向服务架构领域中的一个难题，即架构中多个涉众（用户）的交互。这篇文章正应了这期特刊的基本原则，使用多重视角来解决在特定应用领域的某一类关注。

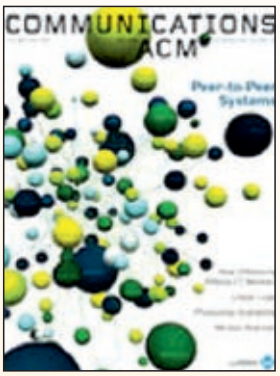
《冲突为中心的软件架构观：质量要求中的权衡取舍》一文认为，许多架构的决定是由冲突驱动的：需求之间的冲突、涉众之间的冲突、实现技术之间的冲突，并描绘了一种基于视角的冲突建模方法。

《企业目标视角》一文提到，在软件工程研究所的质量属性场景的工作基础上，应用了ISO/IEC 42010的视角模板来提出一种视角，以激发和建模涉众的业务相关的目标和制约因素，这往往不属于我们熟悉的功能和质量关注的类别，但在很多架构上施加了相当的影响力。

通过多重视角，处理不同涉众关注的关键原则是匹配合适的成果陈述给每个涉众。在《软件架构理解的可视化工具：一个涉众的视角》中，调查了将不同系统关注可视化的技术。

Communications of the ACM
2010.10

站在志愿者的肩膀上



志愿者对ACM的发展功不可没。2010年的SIGGRAPH年会中，580名志愿者共投入了7万个小时的工作，使得会议顺利进行。ACM的主席Alain Chesnais撰写了一篇题为《ACM，站在志愿者的肩膀上》的文章，他认为协会是建立在许许多多志愿者的能量和奉献的基础上的，表达了对志愿者的感激之情。并且号召更多的志愿者投入到ACM的工作中来。

20世纪后半叶，生物学取得了巨大进展。神经系统科学也被认为是21世纪的主要科研重点，在最近几年里突飞猛进。计算机科学对神经科学的贡献是多方面的、显而易见的：计算机科学给了生物学强大的数据分析工具，产生了生物信息学和基因组学，让人类基因组的测序成为可能；同样，计算机科学是脑成像和神经科学其他分支的核心。而同时，虽然不那么明显，但神经科学也为计算机科学和人工智能提供了新思路和新方法。《计算机视觉的神经方法》介绍了外皮视觉处理理论最近的进展，描述了在多个大脑区域和类神经元单元之间信息处理步骤的模型。接着讨论了视觉皮质的作用，审查了在视觉识别中信息处理的主要计算原则，然后探索遵循这些原则的计算神经科学模型。

《离岸如何影响IT工人》在对离岸外包服务进行研究之后，指出那些需要面对面的互动和到固定地点出勤的工作，很少被离岸外包。而IT工作因为较少涉及到客户沟通以及与固定地点的实体资产的互动，所以IT工人更容易受到离岸外包的冲击。

《私人制造》一文指出，开源的3D打印机不仅允许人们制造以前无法制作的物件，而且也带来了传统制造科技不可能有的全新的设计，因此宣告了新的工业革命的开始。

IEEE Computer
2010.10

可信硬件

本期的《IEEE Computer》包括两篇硬件安全和可信度方面的文章。其中一篇提出了一种新的硬件木马分类方法；另一篇描述了一种新方法，通过自动锁定每个集成电路，保护生产车间中的芯片不受侵权。

另外的几篇文章对欧盟的ICT计划进行了分析，探讨了以就业为目标的课程设计的必要性；还有一篇文章提出了管理处理器资源的基于场景的框架，从而使运行媒体流应用的多核处理器达到要求的服务质量。

由于经济原因，关键系统不可避免地要使用不可信的工厂制造的电子电路。《可信硬件：硬件木马的识别与分类》提出的硬件木马分类方法，为更好地理解现有和潜在的威胁走出了第一步。

防止IC专利侵权的一种有效方法是使得硬件篡改没有利益，而软件破解不可行。《结束集成电路的专利侵权》一文中，通过一种新颖的低成本的方法，将公开密钥的芯片锁定系统和芯片激活协议两者综合，实现了这一点。

《计算职业的未来：读者来信》中探讨了计算作为一种职业将何去何从？

《发展中国家的云计算》一文，提醒发展中国家在抓住并利用云计算带来的机遇的同时，也面临着开放对高级IT基础设施、数据中心和应用程序的访问后随之而来的风险，要最小化这些风险，保护敏感的信息。

《针对有服务质量要求媒体处理的基于场景资源预测》一文中提出媒体流可以被平台无关的场景信息注解，来反映框架级别上的解码复杂度。从而在单核和多核处理器上，实现了节能的解码、资源预测以及服务质量管理。

观点

@theliuyan

昨天在新单位的活动上，重点讨论的就是如何利用互联网、计算机技术及好的设计来提供更好的老年福利产品和服务，因为人口老龄化问题就在当前，如何利用设计和技术让老人主动而独立地参与到信息社会和生活中去，是未来产业发展的一个重要领域。http://bit.ly/cOMs6T

@hengdm

第一不要为老板做产品；第二不要为竞争做产品；第三不要为用户做产品。此为创新产品的基础，坚持产品价值观的必备精神。可惜大多数企业文化很难与此相容，而恰因如此，这是小公司做出革命性产品成为伟大企业的契机。大企业文化的自我保护机制是扼杀创新的凶手，也是让企业走向衰败的推手。

@GossipSama

抄袭又不甘心没创新，做什么都想大而全，不会思考“为什么不”只会“做加法”，简单粗暴堆功能——观“网易八方”及当前“拿来主义”风潮诞生下的各大互联网公司（腾讯、新浪、搜狐、盛大等等）各色软件有感。

@feelinglucky

RT @vvoody: RT @tualatrix: 作为一名项目主管，你知道为什么要尽快地抛弃SVN而转向Git（或HG、Bzr等），答案就在这幅漫画里：http://img.ly/2hjT // 还有应该用脚本语言而非编译语言。

@taweili

Look/feel不能作为控告的基础，Apple在外观上和微软告了10年败诉。看起来像魅族的苦肉计炒作，帮M9造势。RT@kDolphin：抄人家总是要还的。RT@dosou魅族M8因外观上涉嫌窃取苹果的创意和知识产权，被迫停产。魅族这次“被”得不轻。

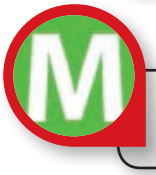
@lidaobing

充血模型和DI有些失配，不过也不是不能解决的，AR 支持raw sql，所以性能反而不是大问题。RT @jeffz_cn：PHP应该去死http://news.cnblogs.com/n/76823/不过里面最让我心有戚戚然的就是我也很不喜欢Active Record。

资源

@tualatrix

在Git Survey 2010中调查“你在使用哪个Git特性”时，“stash”名列第一，约66%的开发者喜欢用它。这恰好是Git的撒手锏功能，其他SCM不具备该特性。https://git.wiki.kernel.org/index.php/GitSurvey2010



@mashabletech

重新定义2010年线上游戏的五个体验。http://mash.to/2Oi4j #MashableAwards

@denharsh

博客新手的15个技巧。http://bit.ly/aRusYn

@newkhonsou

DHH无论写框架做项目办公司都是哲学范儿。搞些震惊业界的东西出来似乎就是为了教化世人举个例子一样…… RT @xu_lele RT @livid: Rework 简体中文翻译版专题小站http://v2ex.appspot.com/rework

文章

@maxchiu

如何在一周内启动一个创业公司。http://t.co/NwQCJYX via @thenextweb

@waynebaby

http://j.mp/aYUwLp 这篇文SSL写得通俗易懂耶！

@PublishingGuru

Google发布网页内分析特性。http://bit.ly/byDHit via @webpronews

@scottgu

我刚刚发表了一篇博客，关于ASP.NET安全性的常见问题。http://bit.ly/bvFvbh

@jurgenappelo

敏捷相关：激励要比才能和流程重要。http://bit.ly/b0MXwZ #agileee

@dannysullivan

如果Google TV 今天被封杀？ http://selnd.com/aDG5AE

（网址链接请访问杂志官网，www.programmer.com.cn）

CSDN最受关注文章 TOP10

（9月26日~10月26日）

1.	程序员：编程给你现实生活带来了哪些坏习惯
2.	中国程序员李洋夺得2010 TopCoder公开赛软件开发比赛冠军
3.	Java之父：我为什么离开Oracle
4.	腾讯这个公司：没有想象力的中国Google
5.	正在消亡的DIY电脑：生意惨淡背后的出路
6.	“程序神童”12小时即可完成手机应用开发
7.	如何在面试中发现优秀程序员
8.	周鸿祎：360需要的，不是打工者
9.	2010年10月编程排行榜：Objective-C挺进年度语言
10.	【分享】给程序员的一些非常不错的资料

JavaEye最受关注文章 TOP10

（9月26日~10月26日）

1.	Doodle 4 Google ：中国孩子画笔下的Google
2.	JQuery 1.4.3发布
3.	【外刊IT评论】PHP将死。何以为继？
4.	AJAXRPC让javascript直接调用服务端方法
5.	淘宝上线独立搜索引擎——淘网
6.	Nutz-1.a.32 发布-尺寸突破800K
7.	Google开发汽车自动驾驶技术
8.	Chrome OS完成RC，确定年内上市
9.	金山词霸涉足即时通讯：似复制腾讯模式
10.	IntelliJ Android IDE 走向开源

CMMI 1.3

主持人：
潘加宇，UMLChina首席专家，潜心研究和实践UML／UP相关技术的应用。

软件工程与管理

10月4日到6日，SEI的CMMI年度研讨会在拉斯维加斯举行，这是SCAMPI主任评估师和CMMI讲师每年一次的交流聚会，今年会议的主要焦点是让主任评估师和讲师们为即将到来的CMMI 1.3做准备。

目前在实施的版本是CMMI 1.2，自2006年8月由SEI发布以来，已经为许多组织的过程改进带来了巨大的推动。CMMI 1.2已经包括开发、采购、服务三个系列，16个核心过程域，降低模型复杂度和提高评估效率是当务之急。SEI计划在11月1日发布CMMI 1.3版本，在12月发布SCAMPI评估方法的1.3版本。CMMI 1.3发布一年之后，将不再进行CMMI 1.2的评估，所有新的评估将基于CMMI 1.3。

虽然草稿没有公开，但从一些评估师在博客和论坛上的发言和讨论可以得知：从版本1.2到1.3，总体的变化趋势是模型更加精简，评估方法更加明确。CMMI 1.3比CMMI 1.2少了将近100页的内容。这几年虽然敏捷的浪潮席卷全世界，CMMI评估依然能够保持着热度。特别是中国大陆，平均每年实施评估的个数超过200家。至今为止，中国大陆通过CMMI评估的组织终于超过了以欧美外包闻名的印度，仅次于美国。不过，评估过程的严谨程度如何，评估质量是否过硬，不得而知，国内论坛上披露的一些乱象让人难以乐观。

SEI也认识到评估质量的问题，所以在2008年和2009年连续发布通知，从严审计所有高成熟度评估，并惩罚违规的主任评估师。今年年初，SEI又发布通知，要求每位主任评估师历年获准实施的SCAMPI方法A类评估的最大数量是12件。不过，市场的要求摆在那里，这样“宏观调控”的后果是主任评估师的要价上涨了，是不是和房地产的调控有点类似？CMMI 1.3中，高成熟度的清晰化是头号重点，目的也是为了减少在高成熟度过程域上的理解偏差，提高评估质量。CMMI 1.3的发布，除了咨询公司准备大展拳脚之外，国内的软件公司如果能够认真研发好一些基于CMMI的过程管理工具，帮助软件组织通过评估，应该很有市场。

说起了CMMI，联想到近年开发社区的一个有趣的现象——“软件工程”、CMMI等字眼成了敏感词？说起这些词汇，总感觉它们太过阳春白雪，虚无缥缈，似乎不同时喊一嗓子“敏捷”以示立场正确都不好意思。但是，容易吃到的午餐已经越来越少，机会驱动、粗放经营就能赚钱的时代已经远去，软件组织免不了要从“出奇”转向“守正”，一些看起来比较严肃的东西，恐怕是再难吃也要吃下去的。所以，软件工程的推动者们有必要理直气壮地说，软件工程是个好东西，不是敏感词；CMMI是个好东西，不是敏感词；建模是个好东西，不是敏感词；软件复用是个好东西，不是敏感词……

本月热点

新品发布

XModelMaker 11发布

这是一款与Delphi和Visual Studio集成开发环境集成的UML建模和重构工具，支持的环境从Delphi 4到Delphi 2010、Delphi XE，从Visual Studio 2003到Visual Studio 2010。

Architexa 2.0发布

新版本增强了协作功能，例如Email共享、团队服务器等。

LMD Systems发布EasyUML

通过文本输入来画UML图，可以通过CVS等管理和跟踪变更。

推荐资源

CMMI 1.3官方信息中心，包括10月5日发布的三份快速参考指南

http://www.sei.cmu.edu/cmmi/tools/cmmiv1-3/

搞笑UML漫画集锦

http://geekswithblogs.net/UlteriorMotiveLounge/category/9120.aspx

主持人：
王翔，软件架构师，主要研究方向为XML、.NET、领域设计和PKI应用。工作之余喜爱旅游、写作和烹饪。

触屏下的数据

数据库

本月热点

●新品发布

Tap Forms 发布1.9.12版本

http://www.tapforms.com/blog/

http://www.tapforms.com/

微软发布SQL Server 2008的SP2，在与SharePoint的报表、信息集成方面 为SQL Server 2008提供更好的支持

InterSystems发布对象数据库CACHE的 2010版本

●事件

EnterpriseDB (Enterprise PostgreSQL 公司)宣布提供对开源版本PostgreSQL 9.0版本的支持，以后开源产品也能够one-click安装

Sys-con.com

http://www.sys-con.com/node/1539113

随着iPhone、iPad的流行，触屏终端环境逐渐作为IT行业的一抹流行色被公众所广泛接受，其中除了苹果公司的iOS外，更为技术人员所青睐的Google Android也随着各种山寨、非山寨的触屏终端系统走入大众视线。

不过三大商用数据库厂商似乎还没有注意到这个市场，SQL Server正在Azure上腾云驾雾、Oracle则更注重企业及企业用户环境，而DB2则在更大的行业信息应用和非结构化信息应用方面防守反击，但对于触屏终端似乎三家都没有涉足。就连iPhone、iPad的婆家也没有提供一个与其产品“门当户对”的数据库软件。不过，即便用最偏激的设想——这类终端最后经过市场淘汰只留下“娱乐消费”、舍弃“商务”，但毕竟也需要信息支持，而且随着用户对于应用、使用体验和信息Mashup方式的进步，终端硬件、终端软件、外部服务和终端数据管理也将进行分化，而后者在Sqllite之外，只有部分小角色偶尔冒冒泡。

早些时候，FileMaker发布了针对iPhone、iPad的数据管理软件FileMaker Go，该产品作为一个Apple App发布，确保用户可以在触屏环境下通过3G、WiFi等链接方式快速浏览和管理FileMaker Pro和FileMaker Server中的数据。Tap Forms也没闲着，近期它发布了1.9.12版，作为一个更像信息分类软件的数据“杂货铺”，Tap Forms恰恰给用户以更便捷的信息使用体验，其中除了对Email、账号、网址、联系人等这些基本信息进行管理外，它还提供1：N的Master-Detail关联关系，这提高了触屏终端的自主信息使用效果。相对Oralce、DB2宣传材料中频频出现的“Enterprise”一词，Tap Forms更多面向“Personal”，这也是触屏终端数据管理系统的普遍“偏锋”所在。此外，作为曾经提供Blackberry、iPhone平台数据库管理软件的RockSolid也发布了iPad平台的数据库管理软件，以往DBA们（包括其他IT人员）为了在办公室外及时发现并诊断数据库的各类问题，往往要“本”不离身，尽管后来有了手机端的管理工具但限于展示和交互能力的不足，这些系统并不能很好满足DBA们远程数据管理的要求，而IPad或其他Android的触屏终端正好实现这个平衡——另一方面满足工作之外的娱乐消费要求，一方面又能在必要的时候更好工作（尽管这听起来更富悲剧色彩）。除了这些相对比较“花哨”的数据管理软件外，作为苹果的官方选择，sqllite不久前也发布了3.7.2版本，这是个“实打实”能够完全在iPad、iPhone上运行的本地数据库平台。

触屏终端设备的普及不仅仅是娱乐、服务平台的普及，其中丰富的交互性也为常连接、偶连接情况下的信息检索和使用提供了便利，大众乐于体验新平台上数据驱动的各类应用，而新的平台上还没有称霸的强者，这个市场还处于诸侯割据前的静夜。

不对称的战争

主持人：
肖新光，网名江海客，安天实验室首席技术架构师，研究方向为反病毒和计算机犯罪取证等。

安全

10月Stuxnet蠕虫成为厂商和各主管部门关注的焦点，Kaspersky和赛门铁克等安全厂商都在VB大会上发布专题报告。这个通过以PC以太网络而最终实现对PLC进行攻击的蠕虫，是7月份出现的。但由于10月分析人员推测伊朗核设施被指是其攻击目标，而其作者被怀疑来自以色列从而引发广泛关注。这是一个传播手段非常丰富的蠕虫，它尝试连接因特网来判断自己是否在内部网络中，然后决定是否通过U盘传播，同时还可以通过MS10-061打印服务漏洞、MS08-067RPC远程执行漏洞、RPC服务、共享服务以及远程访问WinCC系统数据库来传播。

WinCC是西门子出品的工业控制系统，是一个行业标准级别的产品，在国内产品应用范围包括能源、汽车、冶金、交通等诸多行业。而正是WinCC系统环境的专用性给安全厂商构成了一定难题，没有得到西门子或者使用WinCC用户深度配合的厂商，很难完整浮现病毒的全部行为，而被攻击的Step 7 PLC控制工具，则更是安全厂商所不熟悉的。Stuxnet通过替换WinCC系统中的一个DLL文件，劫持Step 7工程文件运行时依赖的库函数。显然对于AVER来说，相关产品都是崭新的名词。

在传统的信息安全体制中，守方所具有的最大优势是其和攻击者之间对于信息系统具有认知的不对称性，在攻击者不断渗透的过程中不对称的天平才开始向攻方倾斜。但随着体系规模的扩大，用户的安全性很多靠安全厂商来实现，而安全产品的通用化和被摊薄的响应成本往往无力支撑高价值目标所要求的安全性。特别是对专有环境，攻方长时间地通过搭建模拟环境进行了解，而安全厂商则对于系统的了解程度既不如攻击者，也不如用户。这就使厂商不能第一时间做出有效响应。我们这里说的响应不是查杀，面对高价值系统的安全问题，查杀是最浅层次的对抗，深度分析、复现效果、评价影响、消除后果、分析动机才是深度对抗的要求。这个层次的对抗，犹如一场战争。

在这次恶意代码疫情中有一个值得关注的细节，就是WinCC的作者把SQL SERVER的用户名和口令直接写在了程序中，而不是作为一个配置文件，这一点就使病毒可以对相关系统畅行无阻。即使是严谨的德国工程师也会违反“代码、数据、配置”三分开的原则，这是因为在过去的若干年内，专用系统基本没有经历过严苛的安全考验，一些陋习没有得到足够的教训。随着物联网时代的到来，PC上走过的一切安全麦城，在物联网上都要经历一次。本月国内用户和媒体似乎更关注所谓的隐私之争，其实这也是一个不对称的话题。这是产品厂商与用户之间的不对称性，从微软是否带有后门、到云计算到底是不是“云算计”，公众理应质疑一切垄断和交互。

本月热点

技术文献

有关Stuxnet蠕虫的相关情况和细节可以参见《对Stuxnet蠕虫攻击工业控制系统事件的综合报告》

最新漏洞

国家信息安全漏洞共享平台（以下简称CNVD）发布最新一期漏洞通报，共收集和整理信息安全漏洞104个，其中高危漏洞23个、中危漏洞5个、低危漏洞76个。上述漏洞中，可利用来实施远程攻击的漏洞有83个，0day漏洞14个。其中包括了linux kernel的多个安全漏洞。

参见：http://www.cnvd.org.cn

由于发布时间关系，上述通报未包括微软的本月补丁，10月微软共发布49个安全补丁，是补丁定时制以来月度补丁数量最高的一次。

主持人：
姚磊，Microsoft Dynamics CRM MCP，多个企业信息化商业解决方案项目经验。熟悉IT规划与需求工程与项目管理。

企业统一通讯，想说爱你不容易

企业应用与行业软件

本月热点

●新品上市

CAXA研发设计PLM解决方案2011系列新品隆重上市

中国最大的CAD和PLM 软件供应商CAXA发布了CAXA研发设计PLM解决方案2011全系列新品，包括CAXA电子图板2011机械版和工艺版、CAXA实体设计2011、CAXA图文档2011和CAXA协同管理2011。

●事件

10月15日，以“智通经纬，博领未来”为主题的IBM 2010年大中华区研发中心开放日在北京召开。就研发如何与市场应用紧密结合、信息技术如何帮助行业转型与产业升级、企业业务模式创新、社会民生进步等多个领域进行深入的交流与讨论。

●会议

中国CRM十周年回顾论坛在京召开

10月10日，由中国生产力促进中心、赛迪网、联合发起的“中国CRM10周年回顾”论坛，在北京希尔顿逸林酒店银杏厅举办。

●推荐资源

Microsoft Lync

Microsoft Lync是Communications Server、Communications Online和Communicator的系列产品的新品牌。该系列现在还包括Lync Web App和Lync Online。Lync 2010和Lync Server 2010的预发布版。

据国外权威调研机构的数据显示，截止到2010年底，企业即时通讯市场规模将达到6.88亿美元。到2011年，即时通讯将取代声音、视频以及文本等成为办公室人群的主要沟通工具。而Gartner也有研究报告表明，未来的企业必将是实时化的组织，企业需要使用即时通讯软件、统一通信平台打造实时化的企业。

统一通讯的价值

“时间就是金钱”这句金玉良言是促成企业采用统一通讯的关键。统一通讯的主要特色之一是能与各种商务应用整合在一起，以往企业员工每次都只能以电子邮件或电话单一通讯媒体的沟通方式，在景气持平时尚称堪用，但在现在这个省一块钱就赚一块钱的市场环境下，企业无不希望透过最少资源达到最大的功效，鉴于此，强调可让使用者同时以多种通讯媒体进行协调沟通的统一通讯，很容易的便成为企业节省工时和通讯交通费用的不二选择。例如业务代表不需要苦守在传真机旁等候客户回传订单，可透过手持行动上网设备接收订单，而生产线的负责人则可透过传真机或电子邮件等接收业务转送过来的订单数据，以利后续生产。

减短信息往返时间，有助于加速管理阶层下达市场决策，而执行单位也能及早获得高层的命令，执行后续动作，如此一来，除企业组织的层级隔阂缩小不少外，亦可降低通讯与差旅等营运费用。

如何规划实施统一通讯？

对绝大多数的企业来说，问题不在于要不要采用统一通讯，而是统一通讯的平台既庞大又复杂，根本就不知从何开始导入。兼容性问题决定统一通讯产品是否普及。建议参考Gartner提出的“企业统一通讯发展蓝图”，企业在部署统一通讯前，应先清查内部有何种通讯设备，并且确定需要经常沟通往来的事业伙伴统一通讯化程度，并订立企业在未来五年对通讯的期许。其后，还要成立一个专职统一通讯发展小组，由这个小组负责连结跨部门的各种资源，并根据各部门所需，订立计划指针与标准作业程序。

市场情况与注意事项

目前，统一通讯领导商不外乎商业软件与网通设备的国际大厂，如微软、思科以及IBM等。由于上述业者专精的领域不一，没有任何一家厂商提供的产品服务可满足企业所有需求。鉴于此，建议企业应将设备兼容列在产品选购的规格要求清单上。因为很多标榜兼容性高的统一通讯软硬件并不兼容于企业用来整合既有通讯媒介的统一通讯平台架构。既然是“统一”通讯，重点在于该产品可紧密整合服务器端至客户端的软硬件设备，无须汰换或荒废既有产品。

电子商务外包，外包企业下一站

企业应用与行业软件

主持人：
邢波涛，现任文思创新TIBCO研发中心资深架构师，负责TIBCO产品ESB（企业服务总线）的研发。关注SaaS管理软件和B2B、B2C电子商务的融合。下一代电子商务和SaaS管理软件积极的实践者。

软件外包在最近几年一直是一个很热门的话题，根据最新的数据，国内最大的外包公司是东软，市场占有率是8.4%，排名第二的是文思，市场占有率是6.2%，软通4.7%，大连华信4.3%，海辉3.8%，总的来看，排名前10的外包公司，市场占有率不到35%。不知名的小公司，软件外包加起来，居然占到了将近65%，换句话说，软件外包还有非常巨大的增长空间，那么这些巨大的空间从何而来呢？

像东软、文思、软通这类大的外包公司，针对国内市场，传统上，各自有各自的地盘。比如：东软以电子政务、参与国家大的项目为主；文思以银行、移动、电信的测试外包为主，软通以ERP咨询实施为主，但针对目前最热的三个领域：云计算、电子商务、移动互联网应用，都还没有怎么大规模进入这些领域。当然主要原因也是这些领域，或者还没有大的公司参与，或者这些领域的公司，还没有怎么成气候，没有成长为较大的公司，引起外包公司的注意。

今年1月，阿里巴巴对国内最大的电子商务服务商——宝尊电商进行了第一轮战略投资，宝尊的主要商业模式之一是在淘宝商城上为众多世界巨头运营品牌旗舰店或专营店，简单地说就是品牌的“网上零售商”，提供一揽子的电子商务服务。这其中包括几十个响当当的品牌。“电子商务市场竞争已经刺刀见红了，而我们绕道而行，进攻的是另一个市场——电子商务服务市场，这个市场在两至三年内可达到2000亿元以上的规模。我们未来的目标是在淘宝网上建一个奥特莱斯。”宝尊创始人仇文彬如是说。“我们的电子商务外包业务已达数亿元规模，收入已达上市标准。”北京兴长信达科技发展有限公司董事长刘磊表示，公司目前服务的客户有诺基亚、摩托罗拉等国际手机品牌。而根据IDC的这份报告，2009年中国离岸外包市场规模也才28亿美元，还不到电子商务外包规模的十分之一，文思2009年全年主营业务收入净额为1.481亿美元。这些数据，与电子商务外包的规模相比，都相距甚远。所以我的判断，电子商务外包应该是传统软件外包企业的下一站，像文思、软通这样的公司，更应该积极并购宝尊、兴长信达这样新兴的电子商务外包公司从而直接进入电子商务外包领域。

电子商务外包的确切定义是什么呢？我这里摘录一个定义：所谓电子商务外包产业，就是在传统电子商务平台之外，为终端客户直接提供客户自己的电子商务平台建站、技术维护、物流、经营推广、客户联络和服务、售后服务等电子商务流程的全套整合外包服务。电子商务外包在美国已经达到千亿元的市场规模，在国内刚刚起步，潜在的市场规模有2000多亿元。看到这个定义，让我想起了另外两家公司：上海商派和万网，他们也都得到了阿里巴巴的直接投资。

本月热点

新品发布

ZK 5.0.4发布，ExtJS 3.3 最终版发布

ZK 5.0.4把主要焦点放在内存优化和引入一些新的特性上，除了内存有重大改善外， ZK 5.0.4 也引入了许多新的特性，例如帧间通信，新的水平垂直布局组建、像slider和combobox的功能的提升。ExtJS最值得关注的有 PivotGrid和很多的日历组件。经常有人再为选择何种Ajax框架而烦恼，我的建议，如果基于Ajax做管理软件或者类似MIS系统开发，首选是ExtJS、次选是Flex、然后就是ZK；而基于Web的纯互联网应用，则优选JQuery。有人说ExtJS太慢，Flex文件太大，“银弹”是不存在的，清官难断家务事，各有各的好与不好，想免费获得“达摩克利斯神剑”，也是不可能的。

事件

德国软件巨头SAP虽然早已开始努力发展网络软件业务，但是在三年的发展历程中，却仅吸引了不足100家用户。SAP幻想先规范、统一中小企业日常管理，然后再大规模实施SaaS管理软件，沿着这个思路做SaaS管理软件，必死无疑。因为SAP是传统ERP软件的既得利益者，它不会主动去革自己的命，只能被动等待别人革他的命。对于那个CRM厂商，用程序员的思维和思路去做联网业务，也是根本行不通的。

主持人：
高昂，资源与环境信息系统国家重点实验室博士生，关注动态语言，喜爱写旅行游记，同时也是OSGeo中国和InfoQ中文站成员。Blog：http://www.gaoang.com/

超越GCC的编译环境—LLVM

编程

本月热点

●新品发布

轻量级Groovy框架发布Gaelyk 0.5.5版本

Gaelyk是专门应用于Google App Engine开发的轻量级Groovy框架，能够辅助Groovy程序员开发基于GAE进行数据存储的轻量级应用程序。Gaelyk 0.5.5版本在URL路由等方面做出了改进。

行为驱动测试工具RSpec 2.0.0版本发布

Ruby行为驱动测试工具RSpec本月发布了2.0.0版本，在RSpec这款Ruby的单元测试框架的 2.0.0版本中，对模块化设计进一步增强，同时增强了测试工具与Rails 3之间更好的兼容性。

Python 3.2发布alpha 3版本

Python于本月发布了3.2 alpha 3版本，这是 Python 3.2 的第三个也是最后一个 alpha 测试版本。对Python 3.2内核、库函数和C-API进行了完善和Bug修补，为下一个Python版本做好准备。

●书籍教程

《LLVM指导教程》

LLVM指导教程提供了LLVM语法分析程序的介绍，解析器和抽象语法树的实现介绍、即时编译器和优化器的支持，以及LLVM的优化教程等内容。

地址：http://llvm.org/docs/tutorial/

C、Objective-C和C++的编译工具集LLVM在10月发布了最新的LLVM 2.8版本。在2.8版本中，LLVM包含了C++标准库实现LibC++，以增强LLVM对于C++规范的支持。新的LibC++库在C++标准符合性以及高效代码生成方面有着很强的优势。同时，在新版本的LLVM中，包含了首次正式发布的新一代高性能调试器LLDB，LLDB应用与LLVM编译器相同的解析器和源代码工具，以替代先前的GDB调试器。

LLVM编译环境是一个集合项目，为面向C或基于C的语言提供新一代高效易用的虚拟机环境。LLVM起源于2000年伊利诺伊大学发起的开源项目，起先主要在学术研究领域中有所应用。在2005年，Apple为LLVM提供了支持以帮助其发展，并保持LLVM与Xcode良好的兼容性，使其成为一个开发者易于使用的编译器。相比较传统的GCC，LLVM的设计更为现代化和模块化。LLVM为开发者提供了中间代码和编译基础设施，并且LLVM能够与IDE紧密地交互和集成，作为IDE的底层支撑基础，为代码补全、重构等功能提供辅助，同时在程序编译、连接、运行环境执行过程中实现全新的编译优化策略。值得一提的是，LLVM还提供了很多编译器以外的辅助工具，如对代码进行静态检查并生成HTML格式的分析报告等。

在LLVM 2.7版本之前，LLVM支持C和Objective-C编译在LLVM 2.7版本发布时引入了对于C++的支持。LLVM的编译器由Clang完成，Clang在编译速度方面非常高效，体现在包括预处理、语法解析、语义分析、抽象语法树生成的各个编译步骤之中。同时Clang在内存占用方面的开销非常经济，较之GCC有大幅度的缩减。与GCC相比，Clang的设计清晰简单、易于理解，具备良好的扩展性，同时能够保持与GCC的兼容性。Clang为程序诊断信息提供了非常好的版式结构，为程序语法错误提供了源码提示以及上下文调用的错误提示。

在程序编译的过程中，GCC完成了从预处理到代码生成的整个过程，其封装式的设计使得很多中间信息无法被其他程序重用。而Clang将编译过程分隔成模块化的阶段，以大幅度增强开发者对于代码的操控能力。

来自伊利诺伊大学香槟分校的Vikram S. Adve，在代码生成和优化大会CGO 2009上的主题演讲《下一代编译器设想》中谈到：下一代编译器、优化器和运行时的基础架构将更注重在静态语言中应用实时编译和动态优化技术，并且编译器将借助更为灵巧的自动调优策略来挖掘程序优化潜能，同时编译器将会采用投机优化的方式来弥补静态分析优化的不足之处。LLVM项目已经在为下一代编译器的实现方向做出努力，我们没有理由不去尝试使用这个致力于超越GCC的新一代编译环境，来为我们的程序编译注入新的动力。

Web Services 之 REST 风格架构设计

编程

主持人：
俞黎敏，IBM高级工程师，满江红开放技术研究组织核心成员。开源爱好者，在各大技术社区为推动开源和敏捷开发积极摇旗呐喊、加油！

随着 Web 2.0 应用的逐渐流行，Web 应用间数据和服务的公开与集成被越来越多的人所重视。REST（Representational State Transfer，表述性状态转移）风格的服务成了很多人的首选，REST可以快速和简便地实现异构应用之间的数据交换，同时可以保证传输的速率和安全性。目前流行的ATOM发布协议就是REST风格Web服务的一种具体实现。

REST风格的Web服务作为传统Web服务的一种替代方式，以其轻量化、易于构建、无状态以及使用HTTP协议等优势受到了很多开发语言以及框架的重视。

表述性状态转移在Web领域已经得到了广泛的接受，是基于SOAP和Web服务描述语言的Web服务的更为简单的替代方法。主流Web 2.0服务提供者对REST的采用是接口设计方面转变的有力说明，这些提供者抛弃了基于SOAP和WSDL的接口，而采用了更易于使用、面向资源的模型来公开自身的服务。

REST定义了一组体系架构原则，可以根据这些原则设计以系统资源为中心的Web服务，包括使用不同语言编写的客户端如何通过HTTP处理和转移资源状态。如果考虑使用它的Web服务的数量，REST近年来已经成为最主要的Web服务设计模型。事实上，REST对Web的影响非常大，由于其使用相当方便，已经逐渐地取代了基于SOAP和WSDL的接口设计风格。

REST这个概念于2000年由Roy Thomas Fielding博士在就读加州大学欧文分校期间、在学术论文“架构风格与基于网络的软件架构设计”首次提出，在论文中对使用Web服务作为分布式计算平台的一系列软件体系结构原则进行了分析，而其提出的REST概念并没有获得现在这么多关注。多年以后的今天，REST的主要框架已经开始出现，并已经被广泛应用于各种平台中。

REST风格的Web服务的具体实现遵循着显式地使用HTTP方法、无状态、公开目录结构式的URI、传输XML与JSON或同时传输的四个基本设计原则。基于REST风格的Web服务的主要特征之一是以遵循RFC 2616定义的协议的方式显示使用HTTP方法。例如，HTTP GET被定义为数据产生方法，旨在由客户端应用程序检索资源以从Web服务器获取数据，或者执行某个查询并预期Web服务器将查找某一组匹配资源，然后使用该资源进行响应。REST要求开发人员显式地使用HTTP方法，并且使用方式与协议定义一致。这个基本REST设计原则建立了创建、读取、更新和删除操作与HTTP 方法之间的一对一映射。

REST已经进入了Java世界的主流，无论是在做SOA架构的开发、做Web服务的开发还是做普通的Web应用，都有必要认真学习REST架构设计和开发方面的知识。

本月热点

新品发布

OpenEJB 3.1.3发布

OpenEJB是一个嵌入式，轻量级EJB 3.0规范的实现。它既可以作为独立的服务器运行，也可以嵌入到Tomcat、JUnit、TestNG、Eclipse、IntelliJ、Maven、Ant和其他任何IDE与应用程序中。OpenEJB被用于Apache Geronimo、IBM WebSphere Application Server CE以及Apple的WebObjects当中。主要的功能特性有，能以独立或者嵌入式等方式来支持EJB 3.0、2.1、2.0、1.1，支持JAX-WS、JMS、J2EE连接器，通过TCP心跳检测以及节点发现来支持多点服务，支持Quartz资源适配器，可以为Tomcat 5/6加入JavaEE 5和EJB 3.0的特性，通过使用JPA来支持CMP，完全支持Glassfish描述符来嵌入式地测试应用程序等等。

推荐资源

Apache OpenEJB
http://openejb.apache.org/

主持人：
涂曙光，微软平台技术专家，专注于.NET、Web、SharePoint、Office等技术领域。现任职于中国惠普有限公司。

彻底拥抱 jQuery

编程

本月热点

●新品发布

Windows Phone 7正式发布

微软在智能手机市场的“搅局”产品，Windows Phone 7，正式发布了。对于开发人员，微软也发布了Windows Phone 7应用开发工具：Windows Phone Developer Tools。

●资源

Windows Phone 7 Developer Training Kit

Windows Phone 7 Developer Training Kit是一整套关于创建Windows Phone 7应用程序的培训课程，其中包含了幻灯片、动手实验、演示等等内容。课程地址：http://channel9.msdn.com/learn/courses/WP7TrainingKit/。

jQuery模板系统简介

在这篇文章中，介绍了本期正文所提到的由微软提供的jQuery Template插件的使用方法。文章地址：http://www.infoq.com/cn/news/2010/10/jQuery-Template。

Data Link：jQuery的数据绑定插件

在这篇文章中，介绍了本期正文所提到的由微软提供的jQuery Data Link插件的使用方法。文章地址：http://www.infoq.com/cn/news/2010/10/DataLink-jQuery。

Microsoft Commits Code to jQuery

在视频中，来自微软ASP.NET产品组的Stephen Walther介绍了微软向jQuery社区所提供的3个插件，并演示了它们的使用。

10月4日，jQuery的作者John Resig在jQuery.com网站上更新了一篇Blog宣布：由微软开发团队创建的3个jQuery插件被接受为jQuery项目正式支持的插件。这意味着这些插件会被持续改进、增强，并保证其在未来与新版本jQuery与jQuery UI库的兼容性。换句话说，所有的Web开发人员都可以在自己的项目中放心使用这些插件，而不用担心它们某一天停止开发、不再更新，不再能与未来版本的jQuery一起工作。

这3个由微软所提供的jQuery插件分别是jQuery Template插件、jQuery Data Link插件和jQuery Globalization插件。jQuery Template插件在JavaScript页面脚本中提供了模板功能，可以将数据对象（数组）绑定到页面元素上；jQuery Data Link插件提供了数据对象与页面元素属性之间的关联，对其中一方的修改会使得被关联的另一方也被自动同步修改；jQuery Globalization插件则为页面脚本提供了超过350个地区语言的全球化支持。在2010年3月份的MIX10大会上，Scott Gu公开宣布了微软将进一步支持jQuery的消息，并同时宣布将为jQuery提交这3个插件的提案。经过了7个月的开发与针对社区反馈的修改之后，这3个插件终于“修得正果”，进入了jQuery正式支持的插件列表。虽然是由微软开发并贡献，但这3个插件的许可完全遵循了jQuery的许可模式，以开源、免费的原则向社区提供。

不出所料，微软的ASP.NET团队正在“彻底”地拥抱jQuery。对于正在学习jQuery或已经在使用jQuery的ASP.NET开发人员来说，这毫无疑问是一个好消息。但要说清楚ASP.NET AJAX的“未来之路”，还真不是一个简单的事情，因为当我们说“ASP.NET AJAX”时，实际上是涵盖了3个东西：ASP.NET AJAX、Ajax Control Toolkit、ASP.NET Ajax Library。

ASP.NET Ajax Library实际上是一个独立于ASP.NET服务器端技术的Ajax脚本库。ASP.NET Ajax Library的想法其实挺好，首先它并不依赖任何ASP.NET服务器端控件，同时也提供了一些很有用的诸如Template、DataView之类的客户端脚本特性。微软可能最开始是想把ASP.NET Ajax Library发展成一个独立的客户端脚本库，但很快发现它的目标与jQuery有极大的冲突，所以ASP.NET Ajax Library的结局不太妙，它面临着没有任何支持、也不会再有更新的局面。

一直以来，jQuery的流行造成了许多ASP.NET开发人员实际上已经在项目中大量使用jQuery的现状。对jQuery的彻底拥抱，使ASP.NET开发人员可以使用一流社区所提供的脚本库，并复用大量现有的jQuery插件。同时可以预见，ASP.NET开发团队会进一步加强ASP.NET/ASP.NET MVC与jQuery的“友好度”，使得创建Web应用程序更高效。

金秋的浏览器

主持人：
钱宏武，职脉网技术合作人，原搜狐互动产品开发部主管，资深互联网社区架构师，垂直搜索领域专家。

Web

火红的枫叶，金黄的麦穗，黄黄的柿子，一切都在提示我们，到了一个收获的季节，对于技术来说，每年这个时候，都是产品重要版本发布的时刻，对于互联网来说，当HTML5基本已经大局已定，那么浏览器就是各个巨头们争夺的重要砝码，从Google，到微软，从Firefox到Opera，都摩拳擦掌。Mozilla 换帅，Firefox新”主人”登场，一般来说，公司要换帅，如果业绩还不错，老大依然想走人的话，只能说明这个公司有更大野心，Firefox在业界的口碑和市场占有率都不错，换一个老大来管理这个在互联网领域举足轻重的公司，估计对于整个Firefox定位还有开发的方向可能都会调整，毕竟下面HTML5的时代，Firefox该走向何方，如何在新的市场中建立起自己的定位，如何在强敌环顾的情况杀出一条血路，新的CEO希望能给我带来更多的惊喜，Google从来都不缺疯狂的想法，首先是提出一个新的图片压缩模式，提出一个有望取代JPEG的新图像格式——WebP，对于互联网来说，最常用的格式是两种JPG和GIF，前者主要用于真彩，有人问过啥是真彩，很简单，就是真实的照片，如人像、风景等，用GIF压缩会比较小，而GIF主要是动画，就是我们看的动画片，那种用GIF压缩小，特别是黑白的片子，而Google提出这种新的压缩方式，其号称对于JPG在同样表现力的情况下，可减少40%文件体积，对于现在很多读图时代，很多网站用这个估计能降低巨大的成本，这种图像的解码和压缩的算法都是基于VP8解码器的图像处理部分，不过这个只能是Google的美好愿望，推广成功的几率并不大，毕竟这种算法只是它一家之标准，太多不想支持它的公司了。而支持这种算法可能就是Google自己的Chrome了，但Chrome硬件加速跳票到9版本，硬件加速是微软IE9浏览器的一项旗舰功能，众多浏览器开发商都急于在新产品中添加这项功能，以加速自己的浏览器产品。对于传统的软件开发者来说，一年推一个版本都是比较快的，看看Google，我想这个才是互联网的开发，所谓诸法揭破，唯快不破，我可以不稳定，但绝对不可以慢。

9月15日，微软举办了一场名为“Beauty of the Web”的IE9测试版发布会，微软表示，相较于以往的IE浏览器，IE9具备如下数个特点：更简约、更流畅的用户界面；更精美的音频、视频渲染；更加人性化的使用体验；更加安全可靠的浏览体验。感觉微软每个浏览器的版本发布的时候，微软都是这么说，但想想微软的推出速度和Google的推出速度，这就是差别，这就是为啥微什么在互联网一直不太如意的原因了

无数的历史告诉我们，垄断就意味着落后，看着现在蓬勃的互联网，上面所有的公司都用自己的力量在建造一个美好的浏览器环境。

本月热点

●每月热点

微软IE9通过97.7%的CSS 2.1测试

据国外媒体报道，近日，微软对外表示，该公司的最新一代浏览器IE9已经全面拥抱CSS2.1支持，通过了大约98%的CSS 2.1测试。

Opera 10.63 正式版：安全性及稳定性更新

更少的Bug，更完善的功能。

Chrome 硬件加速跳票到9版本

据国外媒体报道，与其他浏览器开发商一样，Google也在大力开发利用电脑显卡的全新浏览器加速技术，但用户现在只能等待Chrome 9的发布。

Google 敲定 Chrome 9 的发布日期

主持人：
房燕良，普通程序员成长起来的资深游戏制作人，投身游戏业10年以来从未脱离开发一线。主要代表作品有《仙剑奇侠传3》、《功夫世界》等。

次世代的网游困境

游戏开发

本月热点

●新品发布

空中网《圣魔之血》

空中网3D奇幻网游《圣魔之血》于10月20日开启大规模不删档内测。《圣魔之血》以众神残破遗迹为背景，勾勒出奇幻三界的战乱景象，其四大职业各具特点。角色动作、特效都有不错的表现。

完美时空《神鬼世界》

之前备受关注的暗黑类型网游《神鬼世界》于10月15日正式开启内测。该作是完美时空推出的首款动作游戏，整体画面风格硬朗写实，体现了“暗黑”的风格。

●事件

2010年金翎奖揭晓

10月13日，“金翎奖”2010年优秀游戏评选大赛正式封票，共决出12项年度行业权威大奖。其中，“玩家最喜爱的十大网络游戏”分别为：《传奇世界》、《天龙八部2》、《梦幻西游》、《穿越火线》、《诛仙2》、《绿色征途》、《梦幻迪士尼》、《成吉思汗2》、《剑侠情缘网络版3》、《跑跑卡丁车》。

盛大游戏签约《最终幻想14》

9月16日，盛大游戏在日本TGS游戏展上正式宣布与Square Enix签约，获得《最终幻想14》在中国大陆的独家运营权。盛大游戏董事长兼CEO谭群钊随后在接受记者专访时表示，目前《最终幻想14》推出暂无时间表，收购《最终幻想》团队也尚需时日。

国内首款使用虚幻3引擎开发的MMORPG《神兵传奇》轰轰烈烈地公测了，百度指数达到30万，吸引了120万的注册用户，不过最终测试的情况据说不是很理想。曾几何时，次世代的概念被引入到网游中，很多厂商纷纷宣传自己掌握了次世代技术，并争相购买虚幻3引擎，应用到自己的项目中去。虚幻3这样的高端引擎是否适合国内的网游市场呢？国内的网游是否应该向“次世代”挺进呢？

首先，从技术角度看虚幻3引擎绝对是好东西，功能强大，效果丰富，表现力极强！但是它应用到网游上还需克服一些问题。例如玩家反映的“进入游戏死机大概15分钟，近乎绝望地想要重启电脑的时候，游戏画面出现”，了解虚幻3的程序员都知道，这是引擎在首次运行时编译Shader造成的。可以说虚幻3是一个强大的武器，但想把它运用好也需要强大的功力才行。如果想要很好地把虚幻3引擎应用到网游项目中，我个人认为至少需要花费一个引擎组1~2年的时间。这里前期可以用来学习、掌握引擎的基本用法，然后深入挖掘里面的实现细节。虚幻3引擎毕竟是针对单机的，而且它的首选平台是Xbox 360而非PC，所以还需要很大的二次开发的工作量。之所以说这个工作量很大，是因为引擎本身已经非常庞大，又使用了多线程的复杂架构，可以想象，一开始动一个小地方都是很困难的。据说网易和畅游也购买了虚幻3引擎，并没有急于应用，而是在慢慢研究，这是正确的选择。

其次，必须认识到虚幻3引擎不是万能的。作为一个成熟引擎，它可以说已经优化得相当好了，但是优化得再好的引擎，也需要严格控制资源的使用量，合理设置贴图、骨骼、关卡规模等规格。因为我们做的是网游，玩家是不可控的，所以必须为玩家聚集的情况留出一定的量来。如果只是一味地追求画面漂亮，资源不能很好地控制，最终真的只能是跑在少数“高端”玩家的机器上了。另外，引擎不具备的一些功能也不可强加。例如它本身是针对关卡型单机FPS游戏的，并未提供类似WoW那样无缝世界的功能。而这种大的功能要添加的话，风险是非常高的，毕竟是在那么庞大复杂的基础上做，如果时间不充裕的话，宁愿舍弃。

国内市场对技术升级的接受需要一个循序渐进的过程，并没有因为什么事件使得大家的硬件一下子上了一个台阶。所以，那些照着欧美单机游戏画面水准制作的国内游戏和韩国游戏是有些冒进了，关键是掌握一个节奏。次世代里面一些好的技术我们可以通过消化吸收，视项目的具体情况应用而来，进而提升游戏的品质。但如果想一下子把单机那套次世代技术推向市场，恐怕是不可行的。

Windows Phone 7：帝国反击战

主持人：
马宁，微软最有价值专家，Windows Mobile开发者。

移动

与Nokia在Symbian和MeeGo之间的纠结不同，微软选择了一种“凤凰涅槃”的方式来应对在移动设备领域面对的困境。Windows Phone选择了与Windows Mobile完全不同的策略，Windows Phone只支持一种硬件规格，消灭不同硬件平台之间的差异，不但可以有效降低OEM厂商的研发成本，减少BSP方面的错误，而且有助于开发者更加有效地利用硬件性能，不用考虑对低端设备的兼容性。所以Windows Phone走了一条近似iPhone的路，由于没有Apple那样的硬件销售渠道，所以选择了继续与OEM厂商合作的方式。Windows Phone缩减了很多功能，将重心集中在与用户体验最密切的部分，而对于企业级应用方面，虽然也集成了Exchange邮件、Sharepoint，但数据库和企业级信息系统方面的支持，甚至还不如Windows Mobile 6.5多。接下来还是说说Windows Phone 7前进的方面吧。

虽然达不到iPhone“重新发明手机”的高度，但Windows Phone 7的用户体验的确是下了一番工夫的。Windows Phone 7的UI系统叫做Metro，据说灵感来自地铁系统的指示牌，醒目、简约、时尚，充满后现代风格的艺术气质。看惯了iPhone的水晶按钮后，也许会有一部分人喜欢Windows Phone 7的简约风格。手机的个性化趋势越来越明显，很多感觉是华尔街分析师的报表无法解释的。

微软延续了对于开发者社区的关注，为Windows Phone提供了Silverlight和XNA两种应用开发框架。但不支持C++、多任务，微软在开发者社区方面的强大优势，保证了即使不挖其他手机平台的墙角，也有足够多的Silverlight和XNA开发者为Windows Phone开发应用。

无论如何，Windows Phone 错过了第一时间在中国发售的机会。那么我们是否有机会在中国市场上看到Windows Phone呢？首先要看运营商，如果iPhone的发售权始终限于一家运营商，并且寄予厚望的Android无法撑起高端市场，那么其他运营商未必不会寻找iPhone的替代者，毕竟微软的Marketplace还允许运营商分成呢。当然，这要建立在Windows Phone在欧美市场取得不错销售业绩的基础上。然后就是搬掉三座大山：Facebook、输入法和XBox Live。Facebook不是应用，而是与照相机、联系人进行了深度集成，但对于中国市场来说，谁能够替代Facebook的地位，并且拥有强大的在线服务可以支持Windows Phone？虽然不再开放IME接口，但微软已经着手自己解决中文输入法的问题了，可是XBox Live的问题却不是微软能解决的。智能手机走向云端，已经是无法逆转的趋势了，可是如何监管云端的数据却成了各国的难题，RIM不就是因为加密算法太好而遭到报应了吗。

本月热点

新品发布

9月28日RIM发布黑莓平板PlayBook

RIM在iPhone与Android的夹击之下，在平板市场上发起了反击。这次推出的PlayBook，采用了7寸屏、1G Cortex A9 CPU、1G内存，提供对Open GL、POSIX和HTML5的支持。按这个思路“手机优而平板”，HP手里的WebOS也有这个趋势，看起来我们还会看到很多平板操作系统。

10月6日松下发布Jungle在线游戏平台

松下推出的Jungle掌机是针对MMOPRG类型的游戏开发的，只是不知道松下如何解决MMOPRG的诸多技术问题。虽然，松下对于掌机市场是一个新选手。但第一台次世代主机3DO就来自松下，谁敢小觑这位昔日霸主？

SONY 发布 Internet TV

SONY发布了基于Google TV的Sony Internet TV，希望手机厂商大肆修改Android的趋势，不会发生在Google TV身上。

主持人：
崔海斌/Billy，十几年的开发经验，专注于Java、Android、移动开发等领域。现就职于创新工场旗下的点心团队，就任总架构师。

RIM 新系统借助 PlayBook 现身

移动

本月热点

●事件

乔布斯炮轰RIM和Android，引发众怒

苹果第四财季营收报告不错，乔布斯手握现金，说话也比较硬气起来，最近接连炮轰了RIM和Android，先是说RIM销量已经被苹果超过，绝无可能扳回，然后针对RIM最近推出的PlayBook，说其7寸屏幕过小，属于中间产品，不是真正的平板产品，后来又针对Android的开放，说其不是真正的开放，只是谷歌为掩盖用户整合以及分裂的系统之间差别而放的烟幕弹。对此，无论是RIM还是Android都给予了一定的反击，RIM的回应是说用户会喜欢7寸平板产品，另外RIM的财季和苹果不同才导致销量统计方面落后于iPhone，而Google的Andy Rubin干脆用一段代码回击乔布斯对其开放标准的定义。不过笔者对于乔布斯炮轰RIM的销量问题看法是老乔有点落井下石的意思，而7寸平板产品，还是有一定市场的，当然，所有产品的问题关键还是在于应用是否能够跟得上。

Android在移动开发者中更受欢迎

据IDC于9月28日发布的报告显示，移动开发者对Android手机的兴趣大增。在调查中，72% 的开发者说Android定位极好，将来可以与大量不同类型的联网设备相接，而只有25%的开发者对iOS有类似看法。从长远来看，59% 的开发者倾向于Android系统，而35% 选择了iOS。

对于大多数开发者来说，Android系统上能够有更多的控制能力，而其SDK可以在任何系统上运行，实在是对开发人员非常友好的。

市场调研公司尼尔森10月5日也发布了8月份的报告，称在过去的6个月中，Android 已经超越 iPhone 和 BlackBerry，成为最受欢迎的智能手机操作系统。有32%的智能手机新用户选择了Android，25% 选择了BlackBerry，另外有 26% 选择了iOS。对于Android的开发者一个算是利好的消息是，最新的数据显示：73.8% 的Android用户已经在使用2.1以上版本，2.2版用户比例也达到33.4%，因此大家再开发新的应用的时候，如果针对2.1进行开发，那么就可以覆盖到70% 以上的用户了，而这个比例还在继续增加中。此外，Google的Market在9月底增加了包括香港和台湾在内的29个国家和地区的付费应用支持。Android新版本Gingerbread（3.0）的消息还是没有确定的发布时间，但也有一些模糊的谍照流传出来，与此对应的是一些关于性能和底层触摸接口改进的传闻。而目前多款已经发布或即将发布的基于现有或更早Android版本发布的平板系统笔者并不看好，其最大的问题是应用无法跟上，其次现有的系统底层并未对大屏幕的展现进行相关的优化，相信等到3.0版本发布以后应该才是Android平板系统的最好进入时机。

尽管RIM在9月27日推出的7寸屏的平板电脑PlayBook被乔布斯攻击，但笔者认为还是有其特色的，尤其在用户体验方面得到了非常大的改善，在其全新设计的QNX系统和WebKit的引擎下，同时发布的WebWorks SDK让开发者使用HTML5、Adobe AIR、Flash、Java语言进行开发，而在游戏方面，支持OpenGL的标准也使得我们可以期待一下。相信RIM也会很快推出基于QNX系统的手机系统，唯一的疑问是开发者和用户的接受程度。此外，PlayBook的命名也是和其过往的企业级应用路线有点格格不入，看来RIM是铁心要杀入普通消费市场了。

HP终于在10月20日发布了Palm Pre 2以及新版本的Web OS 2.0，更强的硬件带来更高的性能，以及一些新的用户体验，其基于Web OS 2.0的平板产品也计划于明年第一季度发布。总体来说还是都值得期待的，但对其未来发展依然感觉不是很乐观。

Nokia这段时间不太平，高层动荡人员流失下的Nokia2010大会草草结束，会上只展示了基于Symbian的系统，众所瞩目的MeeGo看来还没有准备好。随后的一段时间内，还传出了Nokia考虑采用Android平台推出几款产品的这种不靠谱的消息。10月22日，伴随Nokia运营利润增加16%的第三季度财报而来的是新执行官挥舞的裁员大棒，此次全球裁员上限是1800人，针对的主要是OVI和Symbian部门。不过对于还坚持在Nokia阵营中的用户和开发者来说，一个好消息是从Symbian^3开始Nokia将会对其发布的手机实行持续的OS升级，而不是之前那种完全不提供新版本升级的服务。

渐进增强的前端优化

主持人：
许阳寅（文河，Twitter：@yyfrankyy），就职于淘宝网，负责搜索产品前端开发，专注于JavaScript的前后端应用，前端性能优化。

Web前端

Web App的流行，越来越炫的技术意味着更复杂的代码和更多的静态资源。随着页面越来越臃肿，交互动作响应越来越慢，前端性能需要更深入的探索。

按需下载和并行下载

现在很多的HTML文档是巨大的，如果把脚本文件放在页面最后，首屏依赖JavaScript的交互逻辑需要等页面下载完，JavaScript文件也下载完并初始化绑定后才完全可用。以前把JavaScript文件建议放在最后的主要原因是<script>标签会阻塞页面资源的渲染以及下载，而如今对JavaScript并行下载的研究比以前更加成熟，通过XHR Eval、XHR Injection、Script in Iframe、Script DOM Element、Script Defer、document.write Script Tag等多种方式实现并行无阻塞加载脚本，从而使得页面各个模块资源得以最大程度地并行下载。

预读取重要资源，缓存常用资源

部分重要的链接及资源文件，通过内容预读取，在用户浏览页面的时候先进行预读取，从而在页面进入下一页面时可以更快响应。

HTML5的prefetch属性

动态创建<object>或者<img>(IE)来发送请求

同时利用本地存储、WebDB、缓存函数计算结果等新老技术，使数据和资源重用，节省计算和请求时间。

更快地响应用户操作

现在越来越多的数据交给JavaScript异步渲染，这种情况下，应用服务器应尽早清空缓冲区（flush early），并将首屏数据作为HTML输出，而不是全部交给JavaScript执行渲染。同时采用数据分块输出，即处理完一段逻辑就输出一次数据，客户端接受到数据立即执行具体的UI逻辑：请求需要的样式，脚本，背景图片，flash等静态资源。这种模式下，服务端和客户端可以并行处理业务逻辑和UI逻辑。这一过程对Facebook提速高达20%~50%。

挖掘JavaScript单线程特性，将数据计算，DOM操作等耗时较长的步骤适当延迟以模拟异步，让浏览器在空闲的时间点做合适的事情，专注于交互最重要的部分，使得响应更为平滑。浏览器环境局限性和复杂性比一般的客户端程序更多，在浏览器单线程处理的条件下，充分挖掘并行特性，让用户更快更平滑地与页面进行交互。

总的来说，前端优化的重点，从教条式的配置优化，逐渐转向跟业务逻辑息息相关的响应速度优化。从用户开始与页面交互，到发送请求，到应用服务器flush出数据，每个中间点的时间消耗都需要精确把控，前端优化无处不在。

本月热点

●新品发布

jQuery Mobile发布1.0alpha

随着移动应用的不断升温，移动领域的前端开发越显重要，这时候jQuery Mobile的1.0alpha已经趋于稳定，给移动开发带来更便捷的体验。

●事件

js1k

js1k是一项比赛项目，参赛者通过JavaScript在1k的代码内写一个程序。大赛前三名，分别创造出梦境般漂亮的动画、真的可以玩的国际象棋和俄罗斯方块。这项比赛充分体现了JavaScript短小精悍的魅力。

Node.js Knockout

Node.js Knockout是另外一项比赛，参赛者使用Node.js在48小时内开发出任何应用，Node.js已经越来越强大！冠军是一个实时的多人游戏，客户端利用了HTML5的WebSocket与Node.js通信，在Chrome和Safari下运行非常顺畅。值得注意的是，前八名当中就有四个实时应用，Node.js定会在实时Web领域大放光芒。

●会议技术

10月 JSConf.eu 在柏林召开

该会议是JSConf.us的一个分会，会议内容包含Stoyan Stefanov的性能优化新模式，Pete LePage（IE团队高级产品经理）的IE9新JS引擎介绍，Paul的高性能浏览器端游戏引擎开发，以及关于使用Node.js开发Server端js应用的各种实践经验。

拥抱移动，拥抱未来

——2010中国移动开发者大会最新报道

■ 文 / 徐蕊

10月21~22日，由全球最大的中文技术社区CSDN和中国最具关注度的全方位创业平台创新工场联合主办、以“迎接万亿移动应用大时代”为主题的“2010中国移动开发者大会”(CMDC，2010 China Mobile Developer Conference)在北京隆重举行。80多位业界领袖、资深人士进行了主题演讲，超过1500位来自一线的应用开发作者、运营和管理资深人士参加了本次盛会，共同讨论移动应用大趋势，分享实践心得与成功案例。

精彩纷呈的主题演讲

第一天的Keynote演讲中，多位业界领袖人物或分析移动大势，或阐述所在机构的移动战略，或解密知名移动应用的成功之道，令与会者获益匪浅。

CSDN董事长蒋涛在开场致辞中表示，移动互联网的潜力空前巨大。不仅用户数会井喷，而且用户的使用率、由此带来的消费意愿也会大大增加。此外，传统的各类IT应用，包括软件和互联网站也将移动化。为此，CSDN将搭建专门针对移动应用的社区，与各领域合作伙伴和广大开发人员一起构建移动生态链，共同迎接万亿移动应用大时代。同日召开的新闻发布会上，CSDN隆重推出为移动开发者量身定制的一个全新开发社区CMDN。这是国内第一个全平台覆盖的移动开发社区，已经与摩托罗拉、三星、微软和北航软件学院结为战略合作伙伴。

创新工场董事长兼首席执行官李开复在演讲中提出，智能手机价格快速下降，应用软件发行渠道多元化，开发者的回报大大增加，这三大动力将使移动应用时代很快到来。而开源软件、云计算使应用的开发和运营门槛大大降低，又为技术人员创造了千载难逢的创业时机。他还建议，创业者考虑项目方向的时候，需要找准用户的痛点，不需要惧怕行业巨头，不要迷恋LBS等热门概念，要参考但不能盲从美日市场，中国移动互联网会经过一个娱乐社交阶段，应用商店模式会被本土化收费模式所取代。

国内的电信运营商在移动大局中的地位举足轻重。中国移动通信研究院院长黄晓庆表示，中移动充分了解目前产业链和开发者目前面临的困难所在，正在通过推出BAE（基于浏览器的应用引擎）、OPhone、Mobile Market、云计算平台和建设反盗版设施等措施予以解决，力争提供一个支持所有客户和用户的开放平台。而中国联通技术部标准管理处处长顾旻霞则首次披露了中国联通uPhone平台的一些细节。这个项目作为国家核高基专项任务，不仅实现了移动互联网安全机制，还将成为联通一站式智能化解决平台的核心，通过自主开发的UNI-CAR中间件和GUI，支持多平台（包括Android/OPhone、iOS、黑莓等）和多编程语言。演讲内容技术细节丰富，引起了很多开发者的浓厚兴趣。

首日的主题演讲者中有三位来自国际顶尖移动平台或终端厂商的负责人。刚刚发布不久的Windows Phone 7是近日业界关注的焦点之一。微软大中华区董事长兼首席执行官梁念坚在演讲中指出，Windows Phone 7是为了配合微软更大的云计算战略而定制的全新系统，因此很多方面都与其他平台不同，对终端的硬件规格和用户体验都有更严格的规定，开发平台和工具力争简易，并为合作伙伴和开发者提供了更多机会。

摩托罗拉副总裁兼移动终端事业部亚洲产品管理总经理沈斌首先分析了中国移动市场的特色：中国用户有短信、手写输入和中文识别等独特需求，对互联网包括新闻和娱乐等的要求也高于欧美，此外，LBS和多媒体服务也将发展迅速。摩托罗拉手机针对这些做了非常多创新，包括更贴心的个性化，更好的社交服务整合等。他还表达了摩托罗拉对中国移动应用和开发者大力投资的决心，除了已推出的智件园之外，还将推出国内首个Android开发大赛。

而RIM中国区总裁谢国睿的演讲则全面解读了智能手机领域赫赫有名的黑莓的在华战略。众所周知，黑莓在企业移动市场占据绝对优势。近年来

CSDN董事长蒋涛号召共同迎接万亿移动应用大时代（左图）
李开复博士、联网时代科技有限公司董事长蔡文胜、资深投资人Mikko Puhakka都赞同年轻一代将主导移动应用的未来（右图）

又发力消费者领域，个人用户已经超过一半，并建立了一个有2200万用户的移动SNS。此外，即将发布的平板Playbook，在移动商店、广告平台、开源应用平台WebWorks、开发工具等方面的投入，都使黑莓开发看去像是一个充满机遇的蓝海。

作为国内最大的电信解决方案供应商，华为在移动应用时代的动向当然引人关注。华为终端公司CMO徐昕泉在演讲中介绍了该公司在移动终端的发展和战略。华为的应用商店智汇云有一大特色，是可以通过许多国家的合作运营商进入其他渠道不易进入的海外市场，这为中国的开发人员提供了新机遇。

百度、腾讯和新浪三大互联网领导性企业都出现在首日大会的主会场，也是大会的一个亮点。有意思的是，他们的演讲中出现最多的字眼都是开放。

百度副总裁王湛详述了公司的移动战略：产品运营以移动搜索为核心，商业模式以搜索广告为核心，市场推广以连锁联盟为核心。他还表示，百度很快会推出开放平台，与广大开发人员合作。百度移动互联网事业部总经理岳国峰在第二天的演讲中则首次发布了非常丰富的搜索引擎获得的移动市场数据，对产品研发人员极具参考价值。

腾讯的移动战略又是怎样的呢？研究院无线中心总监欧阳凯表示，腾讯的整体战略是实现无线有线一体化，把桌面互联网服务延伸移动互联网。具体实施首先是以关系链为中心的内容整合和通信方式整合，其次是以终端整合和服务整合为基础的支撑端到端的基础平台，让开发人员和产品经理可以更加纯粹地关注业务本身。

新浪副总裁兼无线事业部总经理王高飞的演讲当然围绕微博这个热门话题展开。他也公开了许多有意思的微博用户统计数据和一些使用习惯，并介绍了新浪微博的开放平台。由于微博与手机的天然匹配，围绕微博开放应用也是开发者值得关注的机会。

作为主题演讲中唯一的移动应用开发商，Tapulous联合创始人& CEO Bart Decrem的演讲备受关注，他将自己成功运营iPhone应用的经验总结为六项法则：快速发布，高频迭代；多参与行业大会，加强媒体合作；及时引入新应用，产品升级；建立平台，分享应用；做增值服务。

名家论道

主题为“移动应用天使投资”论坛在下午首个开场。李开复、联网时代科技有限公司董事长蔡文胜、资深投资人Mikko Puhakka就移动创业哪些领域有潜力、选择哪个平台等非常实际的话题进行了讨论，不时有激烈的思想碰撞。

李开复表示，年轻一代将主导移动应用的未来，他们应充分利用互联网特性，将娱乐、社交移动应用作为重点去关注。而蔡文胜认为，与十年前互联网创富浪潮类似，现阶段开发人员应该从移动互联网工具入手，获得更多的投资机会。Mikko Puhakka则赞同年轻一代将主导移动应用市场的观点：“当手机成为人们一出生就能拥有的东西时，移动应用的环境和商机就会发生变革。这是需要认真思考的问题。”

下午晚场的论坛“移动应用赢利之道”，由创新工场资深投资经理张亮主持，PopCap游戏公司大中华区经理刘琨、贝宝（PayPal）中国业务部总经理周粼、架势无线CEO叶忻众位嘉宾进行精彩对话，从不同经验与方向对应用作者提出了许多移动致胜策略。

首日的主会场可谓是名家云集，全面涵盖移动产业链。移动产业链中各环节——运营商、终端厂家、平台商、互联网企业、应用开发商和风险投资的业界领袖纷纷发表了主题演讲。晚间，很多开发者兴趣不减，直奔黑莓开发者沙龙、创新工厂沙龙和友盟开发者沙龙而去。

分会场精彩不断，分享成功运营经验和技术秘诀

如果说“2010中国移动开发者大会”首日奉上的是盘“大餐”，那么第二天的五大分会场则可谓“大餐”后的“精致甜点”。10月22日，平台与技

术实践、产品体验与设计、营销与商业模式、移动游戏、创业与投资、走向海外、创业孵化论坛和行业应用八大主题同时开讲。

在平台与技术实现分会场，三星通讯技术研究院bada课题组的欧阳越、播思通讯技术（北京）有限公司产品副总裁伍自坚、Windows Phone 7开发技术顾问王立楠分别就各自的平台进行了详细的介绍，为场下的开发者答疑解惑。盛大创新院资深软件工程师何晓杰认为由于手机功能丰富，导致产品设计出现四点矛盾，功能与配置、配置与电池、内容与展现和布局与体验。针对这些他提出四点优化建议，即UI/UE设计、功能点设计、单独考虑效率、单独考虑电池。风灵创景CEO张磊讲述了做点心操作系统经历的曲折过程。他指出，点心操作系统的成功在于他们发现了用户的需求，即用本地化系统来满足本地化需要。

在营销与商业模式分会场，亿动广告传媒有限公司创始人和首席执行官马良骏、北京百分通联传媒技术有限公司副总裁常龙分别就移动广告未来趋势和市场发展进行了分析。他们认为移动应用广告将是开发者重要的盈利渠道，利用云计算及云存储做精准匹配，是取得良好广告传播效果的必要条件。来自三星Samsung Apps bada项目负责人张琳、天翼空间产品运营总监刘爽、摩托罗拉公司负责中国摩托罗拉软件商店的邵丹分别就各自的软件商店进行了介绍，他们共同为开发者敞开大门，希望更多有志向、有才华的开发者能加入他们的团队。

走向海外分会场，PopCap游戏公司亚太区总裁James Gwertzman的讲座“揭秘《植物大战僵尸》的成功秘诀”立刻将开发者的热情点燃。James Gwertzman分享了这款PopCap最成功、最卖座的游戏《植物大战僵尸》的六大运营心得：（1）设计者要对游戏创新，最好的产品永远是原创的、创新的。（2）让游戏符合众多人的口味，以便适应不同人群的需求。（3）游戏发布后要不断更新。（4）充分利用社交网络，促进游戏销售。（5）游戏必须支持一系列设备。（6）开始时游戏要免费，这样它很容易被嵌入到一些设备中。

此外，PayPal亚太区移动产品总监Rahul Shinghal就“如何帮助开发者赚取第一桶金”的话题展开了讨论。美国硅谷Loop Tek CEO柯博文和WISTONE董事长兼CEO吴刚也分享了在海外市场创业的经验与游戏开发跨平台的经验。

创业与投资分会场，HTC MAGIC Labs研究计划经理张晏诚提出了“内部企业家”的概念。相对于传统企业家，内部企业家若想把一个创新想法转变成产品，就要从实际出发，在公司内承担较大的风险。HTC财务处处长王文渊则就HTC外部投资方向的问题作了阐述。如何提升用户界面质量、如何增加移动平台流量等是公司未来的研究方向。社区网站、电子商务、广告、App Store则是他们关注的领域。139.me副总裁李波针对初创公司在发展过程中可能遇到的制约因素，总结出企业发展要经历的四个阶段，即了解、明确、衡量和调整。他指出，初创公司在发展过程中要不断的将这个循环持续。创新工场创始人兼管理合伙人汪华也给创业者提出五点建议：做好思想准备；找一个有积累的核心团队；在合适的阶段顺势而为；执行比点子重要；全力投入。

在移动游戏专场，来自北京移动之星信息科技有限公司市场总监李靖在“从《杀戮公主》看iPhone游戏设计”的主题演讲中，分享了在iPhone游戏设计领域耕耘的成果。他认为东西方用户差异很大，对于一款游戏的评价大不相同。在App Store上设计游戏，开发者需要注意选好游戏类型；设计好游戏内容、游戏质量以及游戏容量；注重游戏的市场营销和推广。来自成都汉森公司董事长兼总经理贾柯发表了“跨平台 、云游戏”的演讲。他认为，网页游戏向云游戏转变时将全面更新。和传统客户端网络游戏不同，云游戏将从设计理念、服务器端基本结构及客户端处理效率等方面会催生出新的转变。技术可以引领变革，但是价值终究由商业模式所注定。北京掌上极浩公司总经理张涵宇表示，很多Android游戏开发商认为在市场条件不明朗的前提下，游戏开发很难回收成本。但他认为Android游戏和广告模式相结合可以为开发者带来不错的收益。此外，众领域专家齐聚一堂，就“新世代游戏CP的破局之道”展开了激烈的讨论。

最实用的技术与运营经验，火爆的现场，使不少与会者心潮澎湃

拥抱移动，拥抱未来

两天的“2010中国移动开发者大会”，带给与会者一席名副其实的饕餮盛宴，最实用的技术与运营经验，火爆的现场，使不少与会者心潮澎湃。让我们拥抱移动，拥抱未来！来年再会！

IE9意味着什么
——微软IE9 beta发布会观察

■ 记者 / 刘江

微软IE总经理Dean Hachamovitch为观众演示IE9超炫的效果

对于IE9 beta版的推出，全球媒体的关注度甚至超过了以前的正式版。原因很简单，这几年在世界范围内浏览器已经进入了春秋战国时代，从桌面、便携电脑到平板和手机，Firefox、Chrome乃至国内的腾讯、搜狗、UCWeb等，都拥有不小的用户群。而在云计算的大背景下，浏览器作为最重要的入口之一，具有不可替代的战略意义。在这关键时刻，微软是否能固自己既有的优势并转入进攻，全看IE9的表现了。

2005到2006年，互联网格局发生了很大变化，Web标准的改进，Ajax技术的广泛应用，一批全新的Web 2.0网站和应用走入千家万户，这一切微软似乎都没有看到，仍然沉浸在旧日的辉煌里。IE 6中许多与新标准不兼容的地方，使广大Web开发人员怨声载道。而与此同时，竞争对手则在易用性、安全性、标准兼容性、速度和开发工具等方面大幅创新，开始赢得开发人员和极客们的心，IE的市场份额在不断侵蚀。

2006年和2009年相继推出的IE 7和IE 8，尤其是后者，虽然在产品上总体提升明显，标志着微软开始意识到自己的问题，但是时过境迁，与竞争对手相比，IE已不再具有优势地位，总体上仍然不得不处于守势。关于IE市场份额屡创新低的消息不绝于耳。

发布会当天开场前，我在络绎不绝的等待入场的人群中发现了前不久刚来华访问的JavaScript大师Douglas Crockford，还有知名网站Ajaxian的两位创始人Ben Galbraith和Dion Almaer，从那里得到了对IE9非常正面的评价。会场早餐处屏幕上播放的Web发展历程，和本次发布会的主题“释放Web之美”，都无不在透露微软对浏览器的思考。

IE9发布会一开始，微软IE总经理Dean Hachamovitch穿着印着βeauty字样的T-shirt上场，其中的e是IE的图标。他的演讲穿插着许多看上去非常炫的效果，尤其是本来静态的Bing搜索桌面背景却出现波涛排空而来，让在场者都发出了惊呼。IE9的一个主要开发目标，就是充分发掘PC本身比如GPU的潜力，利用底层DirectX，大幅提高浏览器的速度；这方面另一个举措是引入了全新的JavaScript引擎——Chakra。在界面上，IE9 beta比之前的版本更加精简，但仍然保持了一种华丽，总体上与Bing的风格很配；而固定网站消除了桌面与Web的界限，索引标签、单一框和增强选项卡等功能也都是富有特色的创新。对最新Web标准的全面拥抱，以及更多更好的内置开发工具，是IE9的另两个亮点，相信也是开发者最关心的地方。

未来浏览器的目标应该让用户忘记自己的存在，而使Web内容熠熠生辉。Hachamovitch演讲结束，发布会就进入了另一个环节——合作伙伴的Web应用演示，恰合发布会的主题。有来自全球数十个HTML5应用，各具特色，展示了Web技术所蕴含的巨大潜力，可以说是对《连线》杂志Web已死论的最好反驳。中国的数家公司也展示了自己的方案，其中中国国家博物馆虚拟展厅的3D效果获得了很高的关注度。整个发布会简洁明快、重点突出，恰如IE9的界面。

应该说，IE9 beta版的表现非常出色，除了个别指标外，多年来微软第一次全面赶上了竞争对手，而且在界面、标准支持等方面还有更多创新。更为难能可贵的是，从发布会和会后的访谈来看，微软的思路务实而清晰，对未来已经有比较成熟的思考。我问微软IE和Windows Live商务方面的负责人Brian Hall IE是否有Web App Store的计划，他的回答是：Internet就是最好的应用商店。而Windows部门副总裁Tami Reller女士在回答IE9是否有计划支持老版本Windows和其他手机平台时说，IE9是为Windows 7和Vista用户量身定做的，微软希望首先满足核心用户的需求。这种务实、不浮躁在今年3月MIX大会上与Windows Phone 7开发团队的接触中也能感受到。这恰恰是研发成功产品所必备的气度。

微软曾经是浏览器和智能手机系统的王者，但是前几年的自满心态导致产品研发严重滞后，几乎失掉了最重要的两大战略入口。而IE9和Windows Phone 7让人们感觉到，微软又回来了，昔日的锋芒毕露已经改变为沉稳和大气，让人充满期待。

Daniel Sabbah：软件工程的转折点

■ 演讲 / Danniel Sabbah 整理 / 常政

智慧地球，意味着未来的世界越来越感知化、互联化和智能化，这给软件开发人员带来了新的课题：如何迎接复合系统的挑战？IBM Rational软件集团总经理Danniel Sabbah先生在2010 IBM Rational软件创新论坛上提出了一个全新的思路：软件计量经济学。这意味着世界软件工程有望迎来一个新的转折点。

“智慧地球”的概念，相信已众所周知；但是对于这样一个物联网形式的、更加智能化的体系，如何利用好由此带来的设备自由互联的普及趋势，如何处理好其蕴涵的复杂性，还是一个亟待解决的问题。

以医疗保健行业中的“紧急救护服务”场景为例，设想一下，如果患者得了某类心脏病，拥有植入式的设备、传感器，可以通知当地医院心脏病的救急中心，会有什么潜在的好处呢？研究表明，这些植入式的心脏病终端能够减少30%的死亡率。这好比有一个“24×7”的医疗救护人员，在医院里监护你心脏的状况，却没有给医院带来更多的成本。而这样一种医疗设备，里面可能有近20万行的代码，因为它需要和心脏病监护中心交互，同时还必须保证信息是安全的。当心脏科的医生根据患者的反映数据，对患者诊断评估、通知调遣救护车时，可以设想一下这过程中需要的技术： GPS定位、交通路线优化、本地交通管理系统集成、与本地交通状况及时结合等。而急救车通常也比一般的车辆更加复杂，它像一个轮载的数据中心，需要嵌入式的5000万行代码，还有很多软件组件，每个都有10多个不同的接口，去跟踪、追索、升级、测试，此外还要对它维护15~20年。

Danniel Sabbah 指出，软件计量经济学将是软件工程的转折点

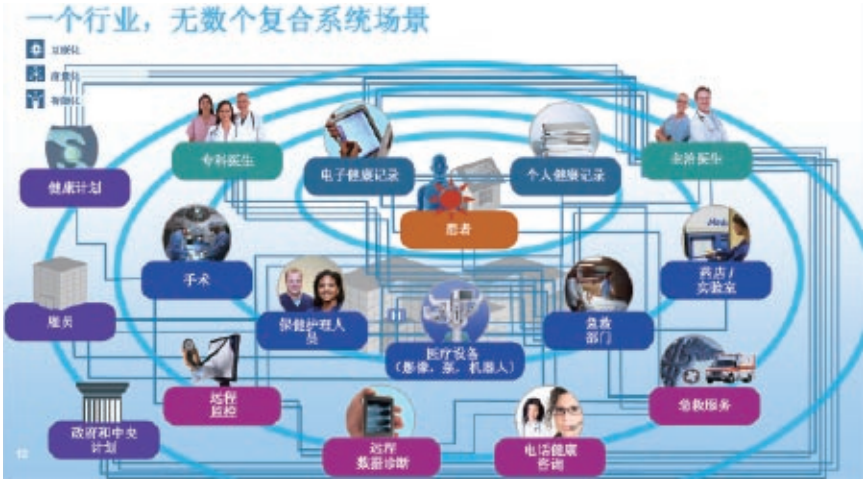
软件工程的挑战

以上涉及的只是预期的最基本结果，是很多行业应用中的一个简单案例。实际上这种“系统的系统”所带来的难题，将会在许多行业中出现，作为软件开发人员是否已经做好迎接这个挑战的准备了呢？

毋庸置疑，我们需要一个转折点——一种更好的途径去构建这些软件和集成的系统，提供预期的经济和社会效益。我们需要一些新的思维、新的技能、新的战略，去应对这些系统复杂性。这里我想推荐一种可以对系统质量进行全新度量的办法，来推动创新——“软件计量经济学”。为什么提出这一个概念？因为是基于经济上的理论、通过经济的原则去衡量企业的价值、衡量给社会带来的价值，能准确地反映出我们创建软件的过程，会帮助我们真正解决未来系统与系统互联的问题，并将效率和质量紧密结合起来，实现一种可度量的创新。

软件计量经济学的原则

软件计量经济学的原则，就是围绕我们社会和经济影响的方方面面的结果，进行货币化，并把这些价值变成一种可用货币衡量的办法和机会。所以，需要开始启动这样一个历程——在传统软件工程不足的地方做起。由于软件系统采用的是传统的工程开发方式，包



传统软件工程模式已无法适应新型复合系统的需求

括计算机辅助的优先处理，还有一些建议系统，比如怎样帮助中心集成商在急救中心使用呼叫工具，还有交通灯抢占的远程控制等。这些在很多情况下会在多个系统之间造成混乱，所以如果不能对结果进行衡量的话，就不能够给它进行优先排序，避免不必要的灾害。比如对于实时的医疗急救例子，如果急救车的调配发生了延误，或者没有找到合适的急救车到达合适的地方去救治合适的患者，我们就损失了时间，也损失了生命。我们不应该只是简单地看每一个单独的系统，而应从一个完全不同的角度看待质量问题。所以创新应该成为整个交付系统的一部分，而不仅仅是开发系统的一部分，比如救护车需要进行优化、减少堵塞，需要和智能交通系统结合在一起。

软件计量经济学的关键要素

为了达到这个目标，我们必须考虑一些关键的因素或问题，使整个交付过程可以实现经济的度量。因此，我们需要从生命周期的角度，进一步改变这一流程的质量和价值，这些都需要在整个供应链中进行衡量，在不同学科和不同系统中进行衡量。结果的风险、成本都必须不断地进行计算，因为没有一个系统会使我们需要有一个静态的要求或在固定的时间里进行静态的交付，这是一个不断循环的系统，而且在不断的发展。我们总是在寻求不断地改善它的行为，而不是一次性的。瀑布式开发方式是不适用的，我们需要规模的灵活性而不是简单的灵活性，是真正的敏捷性、业务方面的灵活性、社会的灵敏性。就像我们最初所说的，我们正从传统的工程模式向新的软件经济过渡，我们必须重新确定它的效率和质量，我们必须重新实现我们经济和社会的效益。为了解决风险、应对风险和降低成本，我们需要动态地不断评估它的不确定性和客户的满意度。系统会不断地实施动态评估，以新的思维不断监督检测这种价值。我们还需要把它与整个开发和交付程序联系在一起，使其成为一个扩展的系统，即复合系统。

为了能够成功地度量这种动态的过程，我们需要一个动态的端对端的生命周期的管理平台，这也是IBM过去30年来一直在研究的工作。目前我们已经开发了一系列开放标准，称之为“开放服务生命周期合作”。这听起来很诱人，叫做OSLC，但是在很多情况下，如果我们忘记了这个诱惑的名称，再考虑一下，像OSLC一样可以完全地了解整个生命周期过程，使得整个平台有了这样动态集成行为和端对端生命周期，并且更重要的是还能够让我们反映和采取行动来获得相应的信息。因为有了信息将会让我们能更好地完成这个过程，把这种演进与变化不断结合起来。这一类平台需要有网络特点，因此我们所做的任何工作，在整个开发平台中做的工作都是依据Web2.0的标准，通过网络合成标准的集成，是整个平台定义的组合部分、不可或缺的部分。我们不仅仅将它叫做基础平台，所有端对端生命周期内的、在协作方面所开发的产品，都可以衡量，有自动化，还有报告等等，这样可以让我们评估、指导和采取行动。比如说Jazz，我们为什么把它叫做Jazz？它这种合作与协调把所有工具结合在一起，来创造美妙的音乐——我们希望能创造出更好的结果。

结束语

总之，我们正在开启一个全新的角度，来探索这样一个以创新和质量为目标的新世界。同时，这也是一个日积月累的、全新的反馈系统，可以让我们持续地关注这一套系统带来的结果，并且进行如实的报告。所以无论是客户，还是我们，都可以共同探索这样一个创新的全新世界。这和IBM的愿景是一致的，即通过更加智慧、动态的业务分析来实现更智慧的系统。也就是说，把Rational软件开发直接与IBM的智慧地球连接在一起。（本文根据Daniel Sabbah 在2010年IBM Rational软件创新论坛上的主题演讲整理。）

可量化的软件价值

——记2010年IBM Rational软件创新论坛

■ 记者 / 常政

每年一度的IBM Rational软件高峰论坛，其所发布的策划、产品已然成为业界了解世界软件工程走向的窗口。而2010年这一届，有超过1500位国内外软件开发领域的创新精英齐聚一堂，尤令人关注的地方是，该论坛名称首次变更为“创新”论坛，同时在IBM Rational 总经理Daniel Sabbath 的主题演讲中，提出“软件计量经济学”新理念。

值得注意的是，在2009年度的论坛中，Daniel Sabbath 就提出过“软件经济学”的理论，它强调客户应该经济地衡量投入产出比，以此指导软件开发中各项工作与资源的配比，以获得最佳的投资回报率。那么这个理论加入“计量”的范畴之后，将有何不同呢？对此Daniel Sabbath解释道，“软件计量经济学”将使得IBM的思维不再局限在软件范围之内，而开始思考电子设备与软件之间的关系。

Daniel Sabbath说：“去年我们谈到软件经济学是把经济测试的标准引进到软件当中，今年我们实际上把这个范围扩大了，不光包括了软件，还包括复杂系统在各方面的产出，也就是说，既包括软件、硬件，还有机械电子的部分，从而能够更好地了解到软件对于复杂系统的产出的综合性影响。我们提供了一个系统分析的方法，可以把结果进行货币化——我们不光能够获得整个复合系统重新组合的价值，而且可以用MPV净现值的概念去更好地通过收集数据进行分析，比如说分析Jazz的平台全生命周期的价值所在。”

IBM认为，软件计量经济学是一个软件交付流程的优化原则，其目标是达到可持续、可度量的创新。为此，IBM特别推出了一种名为“代币”的全新许可模式。据IBM软件集团新兴市场Rational软件总监Mike Rhoads介绍，代币许可模式中的一个令牌包含一定的价值，可反复用于兑换预先确定的功能池中的一组软件许可证。基于该模式，用户可在一定期限内换成不同的产品或用户，无需购买永久软件许可。Rhoads表示：“与传统许可证模式相比，代币许可更经济、更灵活，广大中小企业用户将从中获益。”

IBM Rational中国开发中心总经理严成文畅想Rational软件的未来

伴随着“软件计量经济学”的理念，在本届IBM Rational软件创新论坛上，IBM发布了面向中国市场的三大策略，即“以系统软件开发能力驱动产业升级，以软件园经济加速区域成长，以生态系统建设促进软件产业可持续发展”，助力中国产业的整体创新。与此同时，IBM还发布了一系列具体的解决方案和服务，包括“集成产品管理”解决方案、面向云计算的IBM Rational软件交付服务、最新的Rational行业应用框架等。IBM表示，这些技术举措，将有效地支持客户驾驭软件开发成本与风险，更好地实现产品和服务创新。

谈到Rational软件的未来技术趋势和交付模式，IBM Rational中国开发中心总经理严成文强调了三点：第一，整合的产品管理，从过去的软件交付平台，到现在的软件和系统交付平台，整合的产品管理集成了流程、软件、硬件、服务，从而能够开发更智慧的产品；第二，现在强调的是生命周期管理，这就包括软件和系统生命周期管理；第三，采用更灵活的许可证管理，包括令牌，还有以前固定的和灵活的许可证，以及更上一层的云计算交付方式等。

众说纷纭“软件计量经济学”

■ 记者 / 常政

编者按：软件计量经济学，对于当今软件工程业界还是一个全新的概念。为此本刊记者走访了IBM的技术专家、典型客户。他们的评论来自软件工程、行业需求等角度，您可从中一窥软件计量经济学广阔的应用前景。

软件计量经济学的重要性

Harish Grama：计量经济学为什么重要？为什么我们要衡量？计量经济学的意义，让我们能够优化软件和系统的交付，使它不断地进行衡量、创新和改善你的软件系统的交付过程。记得有本书叫《最佳软件的行为模式》，作者提到当他在与5500个企业交流中有这样一些新发现：首先，有的企业有着非常强的纪律，而且有更好的软件衡量方面的做法，有这样严格的纪律要比那些没有这些纪律的企业更容易成功；其次，他发现只有10%~15%的企业曾经有过有关软件度量的最佳做法；第三个发现，如果你实际上使用这些最佳实践，围绕软件度量和尺度的改进，你所碰到的这些缺陷比那些不使用的少10%，仅仅谈每百万代码行有多少缺陷，这是没有意义的，关键产品里包含的绝对数量，可以说10%即使很少，也会给我们带来一些巨大的影响。

IBM软件集团Rational全球产品开发和支持副总裁Harish Grama

IBM软件集团新兴市场Rational软件总监Mike Rhoads

Kitson&Partners主席Sydney W.Kitson

实施利器：集成产品管理

Mike Rhoads：集成产品管理解决方案集合了IBM及其业务合作伙伴的一系列独特功能、资产和最佳实践方法，将其形成策略，帮助您设计、交付日益复杂的产品，并进行随后的生命周期管理。集成产品管理（IPM）围绕客户的四大迫切业务需要来解决当今最棘手的难题：

- 业务模式不断发展：通过整体业务规划和转型，重塑自身，打造具有竞争力的创新领先企业。
- 产品日益复杂化：产品和系统开发综合了机械、电气、电子、软件技术。
- 企业的外延：在整个设计链上与多个利益相关者开展合作。
- 与经营脱钩：需要管理不断变化的生态系统的资产和经营。集成产品管理是一组功能的集合，不是一件产品。它以一种协调一致、令人信服的方式体现了IBM智慧的产品所要表达的全部信息。集成产品管理包含30多款IBM软件，这些软件都将集中到一个框架内（PDIF产品开发集成框架的演进），在行业销售和支持服务（Industry S&D）中明确将其定位为框架。我们将在智慧的产品解决方案工作组的推动下，继续加强产品集成，并不断添加新的组件。IPM中，Rational内容的一个关键部分是结合了Rational DOORS、Rational Rhapsody、Rational Team Concert及Rational Quality Manager的Rational系统和软件交付工作台。

潜在需求：智慧城市的基础

Sydney W.Kitson：IBM是世界上最受尊崇的技术公司，詹姆威尔是名垂千古的伟大环境保护主义者，但他们都对同一件事做出了断言：“世间万物都是互联的”。因此，我们做的每一件事都影响自己做的另外一件事情。无形的线索数量极大，我们几乎不可能发现并且管理事务中的关联。所以需要通过软件来发现和管理这种关系，软件将真实的世界和虚拟的世界连在一起，以确定打造出智慧的城市。这也是我们Babcock Ranch智慧城市项目和IBM合作的基础，IBM的Rational工具为我们提供了先进、实效的方法来管理各种的变化。

构建一座真正的智慧城市，需要高度重视能源、环境、教育、交通等事务。现在大家实际上对未来之城关心的只有一个主题：一个可以为所有城市互联创造联系的分布式系统，这就是它的核心，不仅仅是数据，而是一种智能分析，Rational工具将会帮助我们确保城市的互联互通、感知度和深入的智慧化，这也是构建智慧城市的核心。

分享与传播：敏捷中国2010

■ 记者 / 高松

Roy Singham认为中国软件将不断持续地出现创新性发展（左图）
Martin Fowler认为从适应性规划转向可预测规划是软件开发过程的精髓（右图）

从2001年敏捷宣言诞生到如今敏捷在全球范围内成为主流的软件开发模式，敏捷已在金融、电信、互联网等行业的项目中落地生根，对软件开发的推广和实践产生了深远影响。

正值敏捷宣言十周年之际，由ThoughtWorks主办、CSDN承办的第五届敏捷软件开发大会“敏捷中国2010”于10月14日在北京新世纪日航酒店拉开序幕。本次大会有敏捷方法大师的谆谆布道，有敏捷专家之间思想的交流碰撞，有中国敏捷实践者的现身说法，短短的两天时间让众多与会者享受了敏捷思想的饕餮盛宴。

开幕式：敏捷大师纷纷亮相

大会开始，ThoughtWorks创始人Roy Singham首先致开幕词。Roy Singham在开幕词中回忆了敏捷方法十多年来的发展历史，他认为中国软件将持续地出现创新性发展，并有机会进入敏捷软件的革命历史当中，且和印度、巴西处于不一样的阶段。

ThoughtWorks首席科学家、世界公认的软件开发权威、敏捷宣言创始人 Martin Fowler就“敏捷软件开发最核心内容是什么、为什么必要”这两个话题分享了精彩观点。在演讲中，他谈到了两个概念：适应性规划和预测性规划。适应性的规划就是在实际开展工作之前进行一个计划，然后依据这个计划付诸实施。适应性的规划忽略了将来可能会发生的事情，没有考虑到客户提出的要求稳定性。而在敏捷软件开发世界里，首先也是进行规划，然后做一部分工作，然后接收这部分工作内容的一些反馈，然后依据反馈再去调整规划，这样可形成一个循环往复的良性周期，即适应性的规划。

在软件开发过程中，从适应性规划转向预测性规划，这种转换是精髓之所在，重点之所在。采用适应性规划，需要新的思维方式，对于软件开发要有一个新的认识。现在要对软件开发采用一个演进式，或者是变革式方法，这些技术方法是开发人员能够掌握敏捷软件开发成功的关键。如果只是关注规划，而不太关注技术，Martin Fowler认为这种方式不是特别好，有的时候会导致失败。

全球精益软件思想与方法的先驱和领袖Mary Poppendieck认为敏捷开发并不是件容易的事情，是一个漫长的旅程，短时间之内很难做到，是循环往复地不断改进。在开发过程中，必须考虑到社会因素，让社会与技术系统无缝衔接，这时必须要知道如何让工作人员和技术系统进行合作。

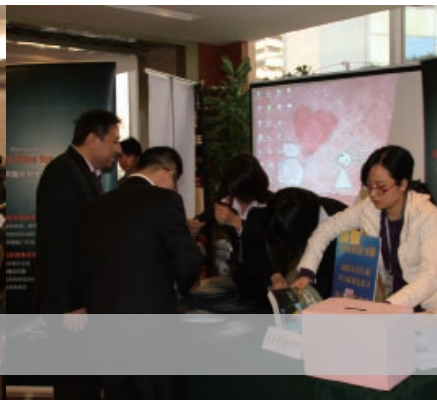
只有让技术元素和社会元素进行优化合作，实现一个平衡，才能实现软件开发的成功。技术人员要在软件开发一开始就关注质量，要有可靠的工艺流程。虽然说员工非常出色，但如果流程很糟糕也会带来很大问题。也就是说，不仅要有很好工作的流程，还要有出色的员工。

分论坛：敏捷专家各抒己见

软件开发工程师、敏捷专家Jean Tabaka与参会者分享了根据她与不同公司合作过程中总结出的成功实现敏捷的12种模式。她表示在实施敏捷时，首先要了解是如何开始敏捷，最好的一个起始方法就是要有一个清晰而具有促动性的目标。再就是需要获得高层的赞助和帮助，这一点是十分重要的。同时在确保整个铺开过程中，要有一个计划良好的框架，明确如何实施一个成功的

Mary说敏捷是循环往复的不断改进的过程


敏捷开发的广泛影响力吸引了众多与会者



敏捷应用。

Renaissance Software Consulting公司创始人、敏捷宣言创始人之一James Grenning带来了主题为“测试不是寻找Bug的游戏”的精彩演讲。专注于嵌入式软件研发的James将敏捷技术引入嵌入式领域，他表示软件是非常脆弱的，新开发的程序代码可能会引入25%的Bug，修复了某个Bug也可能会引入其他更多的Bug，所以要伴随着开发进行不断的测试。如果在项目的最后再进行测试，结果可能会很糟。他指出测试驱动开发帮助软件开发人员提高了开发进度可预见性和产品质量，而这一过程可以通过在写Bug清单或打乱开发计划前消除错误的方法来实现。同时他还表示TDD技术的有效应用不仅可以阻止Bug的生成，更可以帮助程序人员避免开发中常见的无头绪情况。

曾成功促使整个亚马逊推广应用敏捷实践的前亚马逊网络部高级开发经理Alan Atlas表示，软件开发要以客户为导向，团队合作性非常重要。他指出，敏捷之所以可以在亚马逊推广与其独立工作团队的文化有关，他也强调在推行敏捷的过程中，“社区”是一个很好的学习与交流的平台。此外，管理层的理解与支持也是敏捷组织转型的一个重要部分。

无论是ThoughtWorks首席科学家、敏捷宣言创始人Martin Fowler这类大师对软件开发的新思考，还是Mary Poppendieck对精益和敏捷的独特见解，以及中国移动、百度、阿里巴巴、诺基亚西门子的专家实践敏捷过程中的一些体会，都令人受益匪浅。

专访室：国内敏捷正在进行

ThoughtWorks中国区总经理郭晓表示，在中国观察、接受到引入敏捷的过程中，是从流程、管理着手，而忽略了包括重构TBD在内的一些最基本的技术层面实践，跨越敏捷第一阶段，直接进入了第二阶段。由于第一阶段技术方面的缺乏，导致了很大的挑战，在“补课”的过程中，需要把敏捷领域扩展到从开发到部署，在到运营等更大的范围。如果能够迅速把敏捷这三个不同阶段的优点结合在一起，就能找到一条适合中国的道路。郭晓相信，敏捷在国内的发展将比其他国家发展得更快更好。

中国移动通信研究院业务拓展部经理张为民分享他对敏捷研发的心得，包括方式、方法的理解和实践。中国移动研发团队对敏捷宣言中的十二项内容在理解的基础上进行了更新和变换，形成了适应中国移动研发团队的“十二军规”。张为民对通过各实践案例深度解读了这“十二军规”。

在他看来，联动、融合、合作，是互联网时代带来的软件研发的一个变化。大云是一个云计算平台，是互联网时代典型的软件开发案例。在新的时代，研发的方式其实已经与2001年有很大不同，比如研发方法、合作方式、软件代码提交方式等。但是敏捷的精神尤存。在理解这些价值观的时候，会把它继续发展，发展成不同的分支，然后让它最适合业务需求及研发目标。

结语

在敏捷大会后的观众反馈中，看到这样一条：（本次大会）对我莫大的支持和鼓舞，同时带来了很大的压力，我知道自己的责任重大，如何从这场会议上收获得更多，带着这个问题，带着这份责任我迈进了会议大厅。能够将敏捷的思想传播开来，对于一次会议来说是非常大的成功。相信参加这次敏捷大会之后，会有更多的人对敏捷有了一份自己的思考。

网联世界、计算无限
——2010年中国计算机大会记

■ 记者 / 高松

10月11日，为期两天的2010中国计算机大会（2010 CCF China National Computer Conference 2010，CCF CNCC2010）在杭州第一世界大酒店开幕。本次大会以“网联世界、计算无限”为口号，吸引了千余人参会，这是自2003年中国计算机学会（CCF）创办中国计算机大会以来，最为盛大的一次。

计算机产业的中国之声

在大会开幕式上，中国计算机学会理事长李国杰院士致辞并指出：从1999~2009年，中国学者在计算机领域已发表学术论文2万余篇，发表论文总数居世界第2位。近年来，中国学者在国际顶级学术会议和国际期刊上发表的高水平论文明显增加，中国学者需要在国际学术舞台上有更强的声音。

据李国杰透露，2009年中国软件和信息服务业的收入高达9513亿元，增长超过25%，成为仅次于石油开采的第二大高利润行业。一方面云计算、物联网和三网融合的兴起使更多的人相信，传统的通信产业正在向以计算机技术为基础的信息服务业转移；平板电脑等新型网络终端和开源软件的流行也使中国信息技术界期盼自主可控的信息技术基础平台的建立，能从根本上改变我国信息产业的格局。另一方面，2009年，以计算机加工业为主的电子信息制造业销售收入增速比工业平均水平低了8个百分点，增加值率低于工业平均水平5.7个百分点。而且，计算机专业近年来已成为毕业生难找工作的专业之一。正是在这样的背景下，计算机产业应向何处发展、计算机科学技术能从何处取得大的突破，这些宏观性、趋势性问题成为中国计算机界普遍关注的焦点。

大会报告是会议最重要的部分，会议组织者邀请张尧学、怀进鹏、马云、郑纬民、刘海涛、田溯宁、孙凝晖、邓中翰、美国IEEE专家等学界和产业界的专家作了大会报告。

计算无限的云计算

本次大会给人最大的感受是云计算成为大家讨论的焦点，虽然关于云计算的定义、标准等内容，专家们还存在着讨论的空间，甚至每一个企业对于云计算也都有着自己的标准。但是对于“云计算”的时代已经到来，很多人表示相当地认同。

首日的会议上，怀进鹏院士提到云计算时说：“今后将是数据经济时代。谁拥有了大规模的真实运行的数据，并具有有效的处理能力，在互联网中就会成为核心竞争力。”

在会议期间举办7个专题论坛（云计算、搜索、计算机安全、物联网、数据中心、互联网创新、多核技术），其中场场都有嘉宾讨论云计算的话题。

云技术的一个特征是需要建设“数据中心”，有专家就此介绍了大型数据中心的建设与维护，以及MapReduce框架下的Hadoop技术。

计算机安全分会场的专家讨论“云安全”技术，云端数据的安全性是他们关注的焦点。

与云计算同样受到人们关注的是物联网领域，如何将物联网与云计算技术结合起来成为大家讨论的重点。

在如何进行互联网创新的讨论中，也有嘉宾认为基于云计算的服务是未来很重要的创新点。

丰硕的大会成果

本次会议安排了100个IT科学技术成果展览展示：来自全国高校和科研院所的60项国家级科研成果，以实物方式进行演示，有国家重点支持的863计划项目成果、973计划项目成果；有全国著名IT企业的40项重要科技产品和应用的展览。这些展示展览将从多角度多方位反映了中国计算机界的大事件。在CNCC大会上设置IT业的科技成果展览，也是CNCC开办以来的第一次。

与会者在大会晚宴上共同见证并祝贺了中国计算机协会将“2010CCF王选奖”颁给北京大学高文教授和东北大学刘积仁教授；将“2010CCF海外杰出贡献奖”颁给美国俄亥俄州立大学教授张晓东。

大会组委会还向经费困难或西部边远地区的教师或学生提供了特别资助，共有30多名青年教师和学生参加大会。据CCF常务理事、阿里巴巴首席架构师王坚博士介绍，此举是为了中国计算机界更多同仁有机会参加此次交流，帮助更多普通高校的师生开阔眼界，分享信息，创造机会。

用创意和技术摘取Flash大赛桂冠

——Adobe Flash平台应用开发大赛获奖者访谈

■ 记者 / 陈秋歌

为了推广和宣传Flash/Flex技术，同时为个人或者团队提供一个全面展示创意和技术才华的舞台，Adobe公司于6月1日拉开了Adobe Flash平台应用开发大赛的序幕。本次大赛主要面向国内的Flash/Flex技术应用企业、游戏开发者、学生以及Flash技术的爱好者等广大人群。大赛历时五个月，分大赛启动（6月~8月）、网友投票评论（8月）、专家评审（9月）、大赛结果公布（10月）四个阶段举行。

大赛按五个类别分别征集作品和评奖，这五类分别为视频类应用、游戏类应用、移动设备类应用、社交类应用、企业级应用。大赛采用网上报名、网上上传作品形式，网民评委共同参与评选，以百分制为评分原则，分高者即为优胜者，并于10月初公布了获奖名单。本刊记者对其中的5名获奖者进行了专访，请他们分享参赛经验以及大赛心得。

前五名获奖作者及作品对比表

作者	作品名称	作品介绍	所用主要技术
姚卫平	全景漫游者	全景漫游者是一款Flash虚拟全景图漫游制作软件，你可以用它将球型全景地图或户型图、场景音乐和解说、文字信息等各种多媒体元素结合起来，创建出可交互的、专业的Flash虚拟漫游。	Flash 3D
SFGame	物理时空	金甲兔生活在物理时空中，它需要收集空间中的能量才能生存。所以你必须帮助它搭建路径，或者清空路径上的障碍才能使它获得能量。那么就请你来做他的向导吧。	物理引擎Box2D技术
金兆鹏	文锦书苑－手机阅读软件	文锦书苑是一款基于Adobe FlashLite开发的手机在线阅读软件，轻巧省流量。软件具备在线阅读、书签记录、个人收藏、章节跳转、快速阅读、定时闹钟提醒等功能。精心设计的UI界面赏心悦目。	Adobe AIR
吴莹	wing新浪微博AIR客户端	基于Adobe AIR的友好简洁的新浪微博客户端程序。功能更完备：一键操作，博友互动更轻松。内存更节省：挂机微博，轻巧快速不卡机。备份更方便：自动备份，微博数据永不丢。界面更美观：简洁清爽，酷炫特效随心动。	Adobe AIR，Puremvc开发框架
徽炫	狼图腾	一个使用Flash与PS结合制作的实验动画，讲述关于人和狼的故事。	Flash技术

创意来源于生活，返璞归真

本次大赛的参选作品题材丰富，但获奖者普遍反映自己的创意多来源于生活。

“企业级应用”一等奖“全景漫游者”软件的作者姚卫平表示，这款软件的创意来源于南非世界杯、上海世博会对3D全景概念的描述。在设计之前，姚卫平所在的团队对相关市场做了全面调查，发现3D全景软件比较缺少，已有的此类软件不仅功能不完善、定价比较高，而且支持的服务也不到位，很难适合国内中小企业的应用。作为上海八倍信息技术有限公司技术负责人，姚卫平对Flash 3D、高效网络应用技术有着深入的研究。因此他率领团队，根据实际应用的需求，设计了可以将球型全景地图或户型图、场景音乐和解说、文字信息等各种多媒体元素结合起来的、能实现交互且专业的Flash虚拟全景图漫游制作软件。

社交类应用一等奖“wing新浪微博AIR客户端”的作者吴莹，是一名来自江西财经大学软件工程专业的大三学生。在谈及自己的作品创意时，她轻描谈写地说：“原因是为了自己不麻烦”。吴莹是新浪微博的忠实用户，但在发送微博时，每次都需要打开浏览器，这让她觉得非常麻烦，于是就自己动手设计了这款客户端，既方便自己也方便他人。尽管创意来源于很多客户端以及IM（如QQ、blu等），但最初的动机还是完全来自于便利“生活”。

“视频类”二等奖“狼图腾”作者徽炫在阅读了姜戎的《狼图腾》一书后，被狼身上某种灵性的东西所触动，以动画形式给大家展示狼的故事，也算是生活与技术的另一种结合。

文锦书苑——手机阅读软件界面

感受技术突破与分享的乐趣

谈及参加大赛的初衷，姚卫平认为，“希望能与国内广大Flash技术高手有更多交流，提升自己的技术实力，并加强对产品的宣传与推广。”对此很多参赛选手表示赞同。

沈阳的花儿游戏（SFGame）凭借其设计的“物理时空”游戏获得了本次大赛“游戏类应用”一等奖，其作者窦传玲是来自沈阳的一名Flash爱好者，在设计“物理时空”时，用到了关键技术物理引擎Box2D，并用此技术实现了他遇到的一个技术难题——液态球的设计。最终设计出的液体球像水一样，碰到东西有微微震颤，当它足够大的时候，主角能像穿越水滴一样穿到它的体内。

移动设备类二等奖“文锦书苑——手机阅读软件”的作者金兆鹏是一名UI设计和Flash开发者，同时也是《文锦书苑》电子书的作者。大赛一开始他就构思要设计一款针对手机的软件，后来他发现当前看书软件比较受欢迎。经短暂的开发尝试后，他确信通过Flash构建一款手机在线阅读工具完全可行。于是就和搭档联手，搭档负责服务器端的工作，他负责Flash客户端的开发。采访中他说：“如何在机能较弱的移动设备上高效运行Flash应用，是我们遇到的较大难题。”为解决这一难题，他对UI部分进行了优化，同时密切关注内存使用情况，并保证资源的及时释放，尽可能以最小内存开销获得最好效果。这些努力使他们顺利解决了这一难题。他表示这次能获奖也得益于他们一直在真实的手机上运行测试，实机操作帮助他们发现了很多细节上的问题。

关注移动浪潮下Flash技术的发展趋势

采访中，获奖者也谈到了当前Flash技术的发展走向和他们密切关注的问题。

花儿游戏认为便携式电子产品（如3G手机、掌上电脑）上的Flash应用程序可能是未来Flash技术发展的一大方向。窦传玲说：“现在已进入移动互联网时代，移动设备上的应用给人们带来了前所未有的便捷，可以帮助人们随时随地安排自己的生活和工作，比如日程安排、提醒备忘、游戏娱乐、分享见闻等等，这些将成为我们生活中不可缺少的一部分。所以我认为伴随着移动开发浪潮的到来，移动设备上Flash应用程序的开发也将被推向一个新的高点。”

“针对移动平台的AIR是我所密切关注的”，金兆鹏表示他希望AIR能得到更多平台的支持，这样开发者就可以更方便地利用AIR开发跨平台应用，而不必为每个平台单独进行移植，这将是一种非常受欢迎的高效开发方式。吴莹表示她也在密切关注AIR的发展，尤其是AIR在移动设备上的应用，她希望AIR能得到更广泛的推广，完善和加强它在某些技术方面的不足，如组件较大、内存消耗较大等。

作为企业团队技术负责人，姚卫平重点关注的是Flash 3D技术和硬件加速技术。他认为Flash 3D技术的发展离不开硬件加速技术的支持。只有更好地利用显卡等资源的硬件加速，才能提升CPU的效率，使Flash在3D应用领域更上一层楼。

对Flash长期以来的使用者与爱好者徽炫来说，Flash在绘画和动画制作方面的功能应用是他最关注的。比如他希望Flash在绘画上能增加些笔刷的效果，在制作方面能加强动画制作特效功能等。

Adobe Flash平台应用开发大赛获奖名单

视频类应用
一等奖：空缺
二等奖：狼图腾，作者：徽炫
三等奖：罗拉的齿轮，作者：宋建平
游戏类应用
一等奖：物理时空，作者：SFGame
二等奖：阿达坑企鹅，作者：顾方
三等奖：炫点渔夫，作者：李波
移动设备类应用
一等奖：空缺
二等奖：文锦书苑——手机阅读软件 作者：金兆鹏
三等奖：活力英语，作者：王骅
社交类应用
一等奖：wing新浪微博AIR客户端，作者：吴莹
二等奖：新浪微博客户端AIR版本Splus微博 作者：zhuchuanming
三等奖：人魔方网络社交关系搜索1.0.0 Beta 作者：李靖
企业级应用
一等奖：全景漫游者，作者：姚卫平
二等奖：图丫丫——在线图片处理，作者：顾方
三等奖：龙沃在线应用平台，作者：何跃彬
最佳创意奖：阿达点点看，作者：顾方
最佳人气奖：weaverbird编织鸟 作者：李发展

结语

本次Flash平台应用开发大赛，从参赛者的反馈来看，不但汇集了很多国内的Flash技术高手，而且在很大程度上，极大地推动了Flash爱好者们之间的交流与沟通。采访中，无论获奖与否，许多参赛者们普遍希望国内能有更多同类比赛或交流机会，以期在提高自己技术的同时，共同推进Flash技术的进一步应用与发展。

ThoughtWorks技术雷达（2010年8月）

全球知名的IT咨询公司ThoughtWorks的技术咨询委员会定期讨论对软件产业可能产生显著影响的全球技术战略和趋势，从2010年1月开始发布“ThoughtWorks技术雷达白皮书”，有助于业界同仁了解新技术和趋势，更好地应对今天的市场变化。本文是该白皮书最新一期内容的摘译。完整版本请参看《程序员》杂志官网。

技术

企业级软件很少能够在不依赖于其他系统、没有运营团队的支持将系统部署到生产环境、不考虑企业的战略目标的情况下，完全独立开发。但是，根据我们帮助客户团队交付软件的长期经验，却常常看到开发和交付实践中忽视这些问题。

许多系统集成公司会采用一个公共数据库，通过数据库层在应用之间共享数据。许多情况下，这已经成为一种公认而且广泛接受的架构模式：**基于数据库的集成**（**Database Based Integration**）[1]（编者注：数字上标对应图中的符号，其中三角表示新进入雷达范围的技术）。不过该方法存在副作用，会使应用之间的数据库Schema、发布进度、服务的性能和质量的耦合度增加。

DevOp[4]（编者注：开发与运营的合称）是一种新起的运动，目的是满足这样的业务需求：在保证生产环境稳定性的同时快速交付软件。它采用了两种方法：1. 促进开发团队与运营团队的合作；2. 将敏捷实践（协作、自动化、简单等）引入运营过程中，例如变更管理、生产监控等。DevOp涵盖文化、过程和工具。

敏捷软件开发的一个原则是“最后责任时刻”的概念。这个概念在传统的架构师中产生了较大争议。我们认为，如果有正确表述的原则和恰当的测试套件，架构可以不断演化，满足系统不断变更的需求，并且能在最后责任时刻做出架构决策而不损害系统的完整性。我们称这一方法为**演化架构**（**Evolutionary Architecture**）[10]。

跨多个项目的项目要求人员对组织的业务背景、运营模型和战略目标，以及会在计划和设计活动中处理的任何已有的技术、组织和过程限制都有了解。作为企业架构演化方法的一部分，我们采用**业务能力建模**（**Capability Modeling**）[7]为组织需求和目标中核心的业务功能创建轻量级层次模型。能力用目标（做什么）而不是实现细节（怎么做）描述组织的运营模型。

集成的业务流程通常需要横跨数个系统甚至数个企业。新问题出现了：如何协调这些流程？以我们的经验，集中式业务流程编排解决方案经常无法实现承诺，因为实现成本大，而且系统的维护由于需要考虑许多顾客的需求而变得难以扩展。这种情况下我们采用**服务编排**（**Service Choreography**）[8]，其中独立的分布式的参与者按照某个应用协议来协作。基于Web平台，超媒体驱动的应用协议使我们可以实现更易于演化和扩展的集成业务流程。

我们强烈推荐软件交付公司采用**自动化测试**（**Automated Technical Tests**）[6],它包括故障转移测试（Failover Testing）、性能测试（Performance Testing）与浸泡测试（Soak Testing）。这些测试可以开始于项目的生命周期初期，延续到维护阶段。

现在越来越多公司、政府部门与非营利组织开始使用**位置服务**（**Location Based Services**）[60]更有效地与顾客进行交流。随着越来越多地图服务供应商的出现，围绕地图服务的应用也将越来越多。

工具

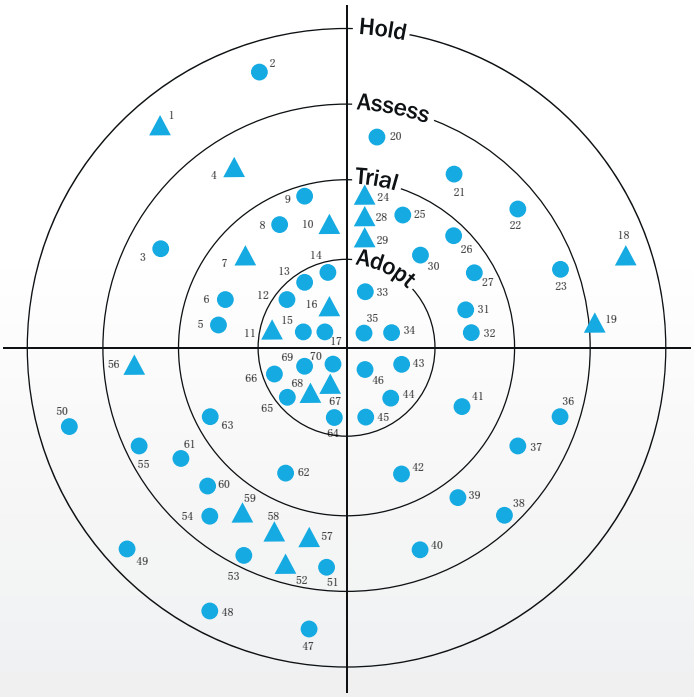
如今所有新项目都需要与原有应用和服务进行整合。结合在系统边缘使用简单的消息总线和集成技术，我们成功应用**Apache Camel**[24]等小型库完成了在此类集成中常见的协议桥接、消息传递、消息路由等任务。Camel具有方便的Java接口、单元测试支持，而且多种不同的传递与消息格式都有连接器，可以在实现分布式应用时提供高效的反腐坏层等。

尽管苹果公司多次公开表示限制**iPhone**[65]和**iPad**[61]的跨平台开发20方案，但仍有很多完全合法的可选方案。PhoneGap和Appcelerator Titanium可以为所有主流移动平台的HTML+CSS+JS Web标准应用提供本地兼容层。

在分布式版本控制工具中，我们推荐**Mercurial**[31]和**Git**[32]。是对于那些愿意与开源社区进行合作的企业，我们推荐**GitHub**[21]。

对**Google Web Toolkit**（**GWT**）[48]我们仍然维持观望评级，但是欢迎讨论。

Puppet[29]是一款免费开源的数据中心自动化工具，它能管理生产与类生产环境的变更。利用Puppet可以对环境配置进行版本控制，以可控制、自动化的方式改变系统。采用像



Puppet这样的自动化工具，可以减少人工操作，将更多精力集中于更高优先级的任务。

NoSQL[26]在大规模数据集、云数据、社交网络数据等方面的处理优于传统SQL数据库。解释何为NoSQL以及建议你是否该选择使用NoSQL是件很困难的事情，因为它包含太广泛的技术：键-值对、列、对象存储以及文档、图和XML数据库等。

使用**Intentional**[19]软件公司的Intentional Domain Workbench最近有少量真实生产环境中的应用，这让我们感到很激动。他们的技术与当前传统软件开发方法有根本区别。现在可以开始探索如何应用他们的技术了。

语言

JavaScript[45]的可维护性、可测试性、可读性等显著特点使开发团队能够方便地开发基于Web的应用程序和网站。JavaScript已经完全足以视为第一类语言了。当然使用时需要应用与其他语言一样的工具（例如单元测试）和方法（例如重构）。V8及其他JavaScript引擎使其性能得到了提升，而在需要富客户端式的体验的领域，Flash和Silverlight在HTML5 + JavaScript面前已经逐渐失去优势。

F#[37]、**Clojure**[39]、**Scala**[38]等函数式语言仍处于评估阶段，但已经相对成熟和可用。函数式语言的两大特性备受瞩目：并行中的不变性及函数作为第一类对象。尽管C#引入了闭包，在后一特性方面具备了部分功能，但不变性才是函数式语言的标志。

F#基于OCaml，现已被Visual Studio工具集完整支持。F#在函数式语言构造之外还非常自然地增加了对象支持及命令式构造。Scala类似于F#，亦糅合了对象和函数式编程方式，但它的语法更像Java。Clojure起初作为**JVM**[64]语言，如今也可以用于.NET CLR。这三种语言存在很多相似之处，但我们认为相较于Scala，F#和Clojure更值得大多数组织评估。

平台

HTML5仍是复杂Web应用程序开发的首选，重要的特性包括增强的音频与视频内容整合能力、客户端存储、更好的文档结构、Web套接字和离线功能。

RDFa[59]是往HTML内容中添加有语义词汇的机制，现正快速而广泛地被内容提供商所接受。这是语义网概念中第一个成为主流的。RDFa支持各种工具，从自定义点集成到使Google爬虫更全面地理解网站内容。如果你想自己的内容能够以简单、廉价、基于标准的方式向大量集成方案开放，建议使用RDFa。我们在看好RDFa的同时，对RDF三元组存储（**RDF triple store**[23]）却保留意见。领先的三元组存储在能力、容量、性能等方面差异很大。如果尝试使用三元组存储，必须进行一系列全面的测试，确认是否符合需要。

另一个Web基础技术平台是**Atom**[68]，它是一种可扩展的数据聚合格式，几乎所有语言都有支持工具。Atom与Atom发布协议相结合，可以组成一个轻量级的、高服务质量保证的信息发布和使用平台。但Atom方案的伸缩性是以延迟为代价的，因此Atom不适用于要求低延迟的方案中。

OAuth[58]是基于Web的授权协议，它允许一个应用访问另一应用中用户的受保护资源而无须共享私密安全凭证。OAuth目前已经是RFC标准，显著提高了Web浏览器和计算机访问分布的Web资源的私密性与安全性。OAuth 2.0将于2010年底发布。由于不能向下兼容OAuth 1.0，而且1.0版本的实现仍然困难，因此我们将OAuth放在评估阶段。

AWS（亚马逊网络服务）是当前最成熟、最广泛的云服务，提供了可扩展的计算服务（**EC2**[62]）、存储服务（**S3**[62]和SBS）、数据库服务（SimpleDB和RDS）、消息服务（SQS和SNS）等。AWS提供的服务列表还在迅速扩展，几乎每月都有新功能上线。AWS平台的优势是否能充分发挥，取决于应用程序。

iPhone改变了移动电话的面貌。iPad则有可能彻底改变用户与基于Web的资源和应用的交互方式，将催生类似平板设备的大量出现。iOS4增加了无线应用发布，使其他公司也能不通过App Store而安全地托管和发布内部应用。iOS4多任务特性打开了企业级应用的可能性，不过这是以增加电池消耗为代价的。

有一类问题如果用增加更多传统硬件的方式解决将非常昂贵，将GPU用于计算为此提供了更好的效率和性能。适合SIMD（单指令流多数据流）处理模型的问题能从该方案获得显著改善，但代价是专门API学习起来比较困难。OpenCL、NVidia的CUDA和微软的DirectCompute都为开发者提供了**GPGPU**[56]（GPU通用计算）的可能。

对话 RSA技术总监 Bill Duane：安全正因云而改变

编者按：Bill Duane于1996年6月加入SDI，也是负责新的代码加密形式的设计师，包括设计智能卡和最新的加密设备。他是RSA的Keon PKI解决方案、RSA著名产品SecurID的设计者之一。

Bill Duane
RSA信息安全有限公司技术总监

安全产业链正发生变化

记者：安全厂商正逐渐受到青睐，比如赛门铁克收购了威瑞信（VeriSign），Intel收购了迈克菲，RSA自身也被EMC所收购，对于安全技术的未来，您的看法是什么？

Bill Duane：独立的安全公司被大公司收购，他们可以获得更多的资本，更多的支持，获得更好的可操作性，应该说对所有人都是有好处的。EMC公司资深副总裁过去几年一直不断表达同一个观点——将来这个世界不会有多少专门独立的安全产品公司，这些公司恐怕都要被大公司收购。可以看到，安全产品的技术研发方式已经改变，现在不管设计什么产品都是一开始就把安全嵌入进去，不再象过去那样将安全产品配套到一个已经完成的产品当中。

安全厂商之间的竞争将是开放和积极的。RSA有一个实验室，这个实验室我们并不要求它赚钱，我们要求它制定标准，确保整个行业都可以在这个标准上持续发展，并且有很高的互操作性，我们相信在云计算的市场上大家都可以获得成功。

记者：那么安全厂商的中立性将逐渐丧失吗？

Bill Duane：坦率地说，这个问题我不知道答案。但就RSA本身来看，迄今为止，EMC并没有阻止RSA和其他的公司包括VMware以外的虚拟化技术公司进行合作，也没有要求改变我们的平台或者我们的解决方案。但其他公司的情况我不太清楚，我们也许可以等着瞧。

安全算法的未来

记者：越来越多的算法已经被包装到了语言和工具中，对于安全算法的研究，我们应该怎样看待它的未来？

Bill Duane：算法是一切的基础，尤其是安全算法和密码算法。我身边有很多人对数学原则有深刻的理解，他们花一生的时间研究算法，在我看来只有这样的人才可以称之为算法专家。但是问题也随之而来，如果这个问题这么棘手的话，这个行业还怎么发展？

安全算法无外乎三种：一是加密算法，二是数字算法，还有一个是Hash算法。好的算法就像坚固的锁一样，但是目前厂商们拿出的系统还没有足够好，所以安全的问题并不在于算法方面，而是要培养足够多的专业人士把好的算法一开始就应用到产品设计当中。如果在一开始我们就考虑如何把安全问题加以解决，要比以后再把安全嵌入到现成产品好很多。

这个行业比较薄弱的环节仍然是如何把好的算法应用到产品和系统当中去。RSA在上海有一个硬件高级开发小组，我是这个小组的负责人，我有三个工程师，他们的工作就是开发安全产品并且拓展新的领域，我给工程师的指示就是他们要尽可能了解学习密码算法的知识，并且了解如何应用密码算法，但是我并不期待着他们本人开发出很好的算法，只需要他们可以把这个算法应用到安全产品当中，从开发算法的角度来说，我们有非常好的数学家、密码学家，他们都可以帮忙做这些工作。

记者：RSA生成存储密钥的速度过慢多年来一直被开发人员所抱怨，用超宽处理器是唯一的解决方案吗？成本是否太高了？

Bill Duane：的确，RSA公司的算法是CPU密集型的，加密和解密非常方便，但产生的密钥也非常大，需要更长的时间。

在这个行业发展的过程中，老有人说RSA算法快不行了，但是处理器的速度加快了，因此RSA又得救了。作为技术人员，我们认为RSA的算法不会被别人取代，至于现在密钥的生成时间长的问题，不一定会由RSA公司解决，一是处理器的速度更快了，再有处理器现在都会嵌入生成密钥的硬件，比如英特尔移动平台上有加密的相关功能，这样在内部本身就可以加快生成密钥的速度。再有，这些硬

件的随机生成工具，并不是专门针对RSA公司生产的，它只是随机生成一些对称的ECC密钥，是一个非常基本的功能，但是它对RSA公司来说是有好处并且有用的。

云计算挑战安全技术

记者：在云环境下，我们的PKI将会发生什么变化？

Bill Duane：在云计算的框架下我们会遇到一个现象，就是多租户同时存在，因为我们把本地的架构虚拟化了，可能多个租户都会使用同一个硬件，在过去可能这些用户不会出现在同一个硬件上，可现在这种情况却出现了。

多租户会产生问题，比如某个应用在虚拟机上运行，运行结束之后退出，但是数据在内存上存在过，另外一个新的虚拟机进来，因为他们使用了同一个硬件，使用了内存和硬盘，在这种情况下就会产生信息的泄露。另外一个是侧面攻击，我们知道虚拟机应用硬件的时候，可能会使用同一个CPU，我们在多租户的情况下，可能就有理由担心会产生信息的泄露，现在其实有一些技术，比如说清除内存或者抵御侧面攻击的技术，但是这些技术要应用到云计算的环境下显然还需要进一步的改进。

此外，在用户把数据迁移到云的架构当中的时候，用户也要知道这个供应商是否有足够的技术保证数据的安全。据我所知，复旦大学就有一个项目，他们正在研究如何从硬件到应用层建立一个可信任的链条。另外一个非常重要的领域就是法规的遵从，我们知道在应用相关服务的时候，要确保有关应用符合政府的法律法规或者行业规定。但是在云服务的情况下是由其他的公司或者其他的企业来运行有关的数据，这样你就需要与云服务的供应商合作，确保他们所提供的服务使得你这种应用仍然是服务法律法规，符合法规遵从。

记者：传统的身份认证技术，会因为云的出现而改变吗？

Bill Duane：身份认证技术应该说现在面临一个非常有趣的时刻。随着移动终端的增加，游戏机的增加，身份认证的应用会更频繁。对最终的用户进行PKI认证非常困难，不像对服务器认证那么成功，但是现在每一台游戏机、手机本身都嵌入了用户的认证技术，也就是PKI。PKI本身是非对称的加密方法，它更适用于整个生命周期的认证，但是对于一次性的密码认证来说，是一种一对一的认证关系，效率可能更高一些。对称和非对称的两种认证技术将来都会存在，也会有这种公共钥匙的技术存在，传统身份认证技术在云服务应用以及在更多与移动终端应用情况下会获得新的生命。

移动终端和计算机的联系将会越来越多，手机本身是存在身份认证的，这些身份认证技术也越来越支持移动的终端。另一方面，移动终端本身是个认证器，帮助人们进入整个IT环境，比如我把自己的手机输入密码进行了身份认证，通过蓝牙可以把这个认证发给我的电脑，因为双向的身份认证是可以通过的。换句话说移动终端有两面性，一个是平台，另一个是身份认证信息的存储器，在这两者之间并不是一个选择的关系，而是并存的关系。

记者：安全厂商对于硬件安全的关注度还在持续上升吗？

Bill Duane：从RSA内部来看，硬件人员和软件人员的比例上硬件还是占少数，但是我们的确在加大对硬件安全的关注度和投资。RSA和英特尔有底层硬件的合作，和威瑞信也进行了一些硬件方面的合作，比如手机平台的解决方案等。6个月以前，RSA在上海成立了硬件安全开发小组，硬件小组的任务有三个，第一是技术拓展，开发新的一次性密码卡，第二是拓展我们的安全技术能力，第三是确保EMC平台的安全。

移动安全的新热点

记者：手机支付的安全问题会是一个新热点吗？

Bill Duane：这是必然的。RSA有各种各样的产品可以提供一定的帮助，从比较低的层次上来说，我们有加密的算法，这种加密的算法可以应用起来，使得基础的支付系统是安全的。在更高的层次，我们还有法规遵从方面的一些产品，还有事件监控的产品，这些产品也可以组合成相应的系统，跟踪相关的交易，查询交易的记录，看一看这些交易是否出现了偏差，看它是否符合监管的要求。另外RSA还有防止数据泄露的套件产品，可以保证信息在发送的时候不会泄露个人隐私。

另外RSA一开始就有一个产品，主要是为了有效地管理一大堆的密钥，它现在已经发展成为一个系统，可以管理很多需要保安的物件，在你需要的时候提供给你。

当然了，RSA不会直接生产这种产品，我们会更多提供不同的组件，由其他厂商设计出相应的解决方案。

记者：但是我们所考虑的好像不仅仅是技术问题。

Bill Duane：我们确实看到世界上其他国家在确保支付系统安全解决方案的时候都有不同的方案，换句话说安全技术或者支付技术并不是手机支付能否获得成功的关键，还有其他的因素，比如经济环境、监管等等都会影响到解决方案的设计，作为一个安全系统的架构人员，从安全的解决方案来看，希望他们不要失去安全性，不要暴露用户的隐私，这两者很重要。但是我们一定要知道，在这方面政治因素和监管因素不应该成为对技术产生很大影响的主要因素。（记者/谭茂）

26岁的 CEO

大学时，他开始着手创业，现是北京三记电子商务有限公司CEO；同时，他又是北京首都国际机场BOWEI公司高级软件工程师，主掌行李控制中心核心数据库；未来，他将投身于旅游电子商务事业，一切都在按部就班地进行中……熊丹，26岁，已经完成了从CTO向CEO的转变。

熊丹 北京三记电子商务有限公司CEO

记者：从北京航空航天大学到自己创立公司，再到北京首都国际机场BOWEI（下称“BOWEI”）公司，在这八年的努力中，你感触最深的是什么？

熊丹：在内心，我认为学业和职业是不可分割的。无论何时人都需要不断的学习和创新。

大学时，我认识了一些能和我一起创业的兄弟。从完全不懂IT到迈进这个领域，他们处处帮助我，给予鼓励和关心，这让我能全心思地投入在钻研中。

在工作中，BOWEI高层领导及部门经理对我工作的支持和理解让我感受颇多。对于一些关键项目，他们充分相信我的能力，不施任何压力让我自由发挥。从项目周期、人员调配和时间安排上，都尽量满足我的需求，这些帮助我都默默记在心里。

记者：从大学到现在，你参与过多个关于“IT人才认证”的考试。对于IT人员而言，他们应该怎样权衡能力培养和证书、学历的培养？

熊丹：我认为IT人员的素质不在于证书的多少，而真正在于他们是否有能力去做。我考取证书除兴趣外，就是要证明自己。我建议IT从业者不要把证书看得过重，自身能力的培养和工作水平才是根本。技术不断在更新，永远有一些新的知识需要我们去学习和完善。

记者：在你的两个身份中（CEO和高级软件工程师），你更偏重于哪个？这么多年的努力，你的动力来源于哪里？

熊丹：我更偏重高级软件工程师的身份，因为我一直保持60%的精力在这里。

首都机场的行李系统和旅客捷运系统是支持机场运行的两大核心系统，我在这其中负责行李系统的核心数据库。如果“后台运行的支撑业务”出现了问题，后果是不可估量的。几年下来，我深刻地体会到“责任+信任”的力量。这个项目是好几百人几年的心血，我有幸能从头到尾参与其中，有什么理由不把重心放在这里呢？

记者：仅仅26岁你就做出许多成就，有何感想和经验分享给他人？

熊丹：这些年，我学到了很多，也积累了很多。对于那些正在创业的兄弟们，我要告诫他们如果决定了就要坚持拼下去，成功总会到来。因为只有做过，才知道自己离成功有多远，才会积累到属于自己的“财富”。

今后，我会在经济学和管理学方面继续研究下去。在技术领域，我会永远保持学习的心态。我的目标是“不用最牛，但能解决问题就好”。

记者：你认为中国IT技术人员应该在哪些方面加强？

熊丹：我认为中国技术牛人很多，他们最应该注意提高个人修养和外表形象。技术人员已经不再是埋头苦干的“民工”了，他们要懂得包装自己，将自己推销出去；此外，技术人员要足够自信，多和外界沟通，把自己的技术和权威大声地说出来，不要只停留在那些“技术文档”上。我相信一个有自信的能沟通的技术人员才是真正的“牛”人。

记者：你认为一位合格的CTO应该具备哪些素质？

熊丹：除了过硬的技术外，心态要保持冷静。

CTO是一种特殊的职能岗位，企业要求他们对技术和管理都要精通。我认为高级软件工程师应该在行业里享有权威的地位。放眼看看成功的CEO，很多不也是从CTO转变过来的吗？我也不例外。

但对于他们来说，管理技能也是必不可少的。管理是科学，技术亦是科学。所以，第一，在技术上要保持一颗学习的心；第二，遇到逆境要有一颗坚持的心。

我的个人经验是：不管是在管理团队或是从事技术工作，充分沟通才能保证团队血液的良性循环。

记者：你认为当下的技术创新趋势是什么？

熊丹：从传统互联网，到形态各异的移动互联网，这期间造就了许多行业精英。以后的技术创新是基于移动互联网（3G）为媒介的，加之国家这些年力推的三网融合、物联网，这些都是新技术创新的表现，相信会造就出新的商机和产业链。（记者/徐蕊）

幽　默

盗梦的计算机理论：

《盗梦空间》的主要思想是：如果你在虚拟机里的虚拟机里的虚拟机里安装了一个虚拟机，那么所有的东西都会变得死慢。

《盗梦空间》就是一层一层的函数调用，在内存中表现为一个栈结构，如果任务失败，堆栈溢出，就被扔到迷失域，等着被垃圾收集吧！

一句话幽默

老婆问什么叫“开光”，我想想说，大概就是“激活”吧……

一位程序员的网名1024，我一直以为他是很闷的那种人，那天去他公司，看到他们老板的门牌是768。

一位同学一直在复习计算机考试，某天踢足球，另一同学带球到了底线，只听他大喊：回车！回车！（传中）

我的姐夫是一位计算机迷。有一天，我的姐姐从商店转了一圈后回到家里，把她新买的那件睡衣举了起来，要姐夫评价。我的姐夫回答道：“好漂亮的软件！”

iPhone手机有一新版软件听声音能辨别西瓜是否成熟，朋友拍拍自己肚皮录了声音做分析，分析结果是她的肚子已经熟了。

如果你的朋友很久没有和你联系，一种情况是他死了，另一种情况他是读计算机的。

只有这种没有女友的IT宅男才会去研究HTML。

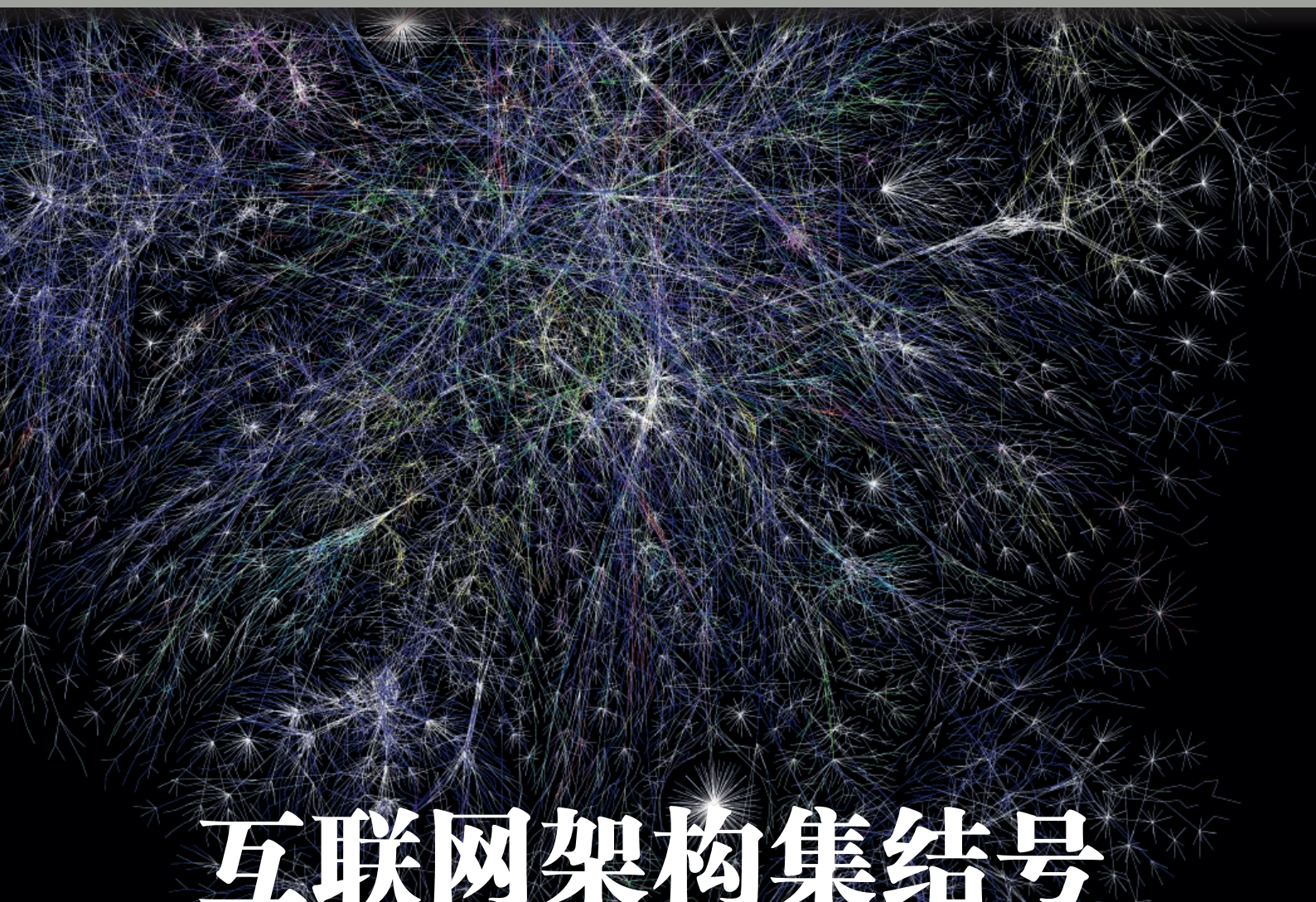
天堂与地狱

一位程序员突然发现自己站在一个审判庭上，需要自己决定是上天堂还是下地狱。庭审委员会说他可以发表自己的意见，还问他是否想先去天堂或地狱参观一下。

“当然，”程序员回答道，“天堂是什么样我很清楚，让我看看地狱吧。”于是一位天使把他带到了一个阳光明媚的海滩上，成群穿着比基尼的美女在打沙滩排球、听音乐，十分享受。

程序员惊叫道，“地狱看起来太棒了！我要去地狱！”却立即发现自己被一群小鬼撕咬着丢进了一口滚烫的油锅里。“海滩呢？音乐呢？美女呢？”程序员对着天使发狂地尖叫起来。

“刚才的是个原型。”天使回答道。



互联网架构集结号

■ 策划 / 本刊编辑部

今天，人类社会的信息基础设施可以说完全建立在互联网之上，从老百姓的日常生活，到企业内部的管理系统，从电子商务、媒体到政府的运作，都已经离不开互联网。互联网从诞生到现在，网站的规模不断扩大，存储和处理的数据量也远远超出了人们的想象，近年来又出现了对信息实时性、多媒体需求大幅增长的现象，互联网架构面临越来越大的挑战。

为了应对这些困难，互联网架构领域也出现了许多新动向：LAMP等开源技术成为主流并开始承担关键任务，越来越多的网站开始采用NoSQL为代表的分布式存储和MapReduce为代表的分布式计算，Restful架构风格、消息队列、数据仓库、负载均衡等技术得到了更多关注。

本期封面报道，我们特邀请了来自知名互联网企业和技术厂商的众多资深架构师，为您吹响互联网架构的集结号，全方位分析互联网架构方面的热门技术和实战经验。

互联网架构一席谈

——知名架构师畅谈互联网架构热点话题

■ 记者 / 董世晓

为了帮助大家准确把握互联网架构的热点，本刊记者特别采访了国内五位知名架构师，详细解读互联网架构的现状及趋势。

互联网架构中，最大的热点有哪些？为什么会成为热点？

崔金峰：两个方面：第一是海量存储，包括大数据量存储，一般就是分布式存储的解决方案；第二是海量计算，主要是用分布式计算。

这些之所以成为热点主要是因为Web2.0以后的时代，更关心个人信息方面的存储和计算。比如Facebook、Twitter等。当这些信息量存储足够大，要想分析数据或者实时提取数据，分布式计算技术必然要应用上。比如类Hadoop的典型应用。

邓毅：第一是大规模、低成本、高性能的存储与运算框架；第二是友好的界面支持、AJAX/HTML5等。

冯大辉：NoSQL相关的解决方案越来越多，与存储和RDBMS相关的挑战已经不像几年前那么严峻。随之而来的是SNS应用中日益严重的“消息处理”需求而带来的压力。用户对信息（尤其是碎片化信息）的及时性要求较从前有了更高要求，目前还看不到更好的解决办法。

崔金峰，58同城网副总裁

侯震宇：互联网架构是个很大的范畴，涉及方方面面，自然热点也是方方面面。如果要寻其热点，我们要首先找到互联网公司都有哪些类型，不同的公司又遇到了哪些架构层面的问题。

我把这些公司按照现实社会也分成三个世界：发展中公司、发达公司、超级公司。他们面临的架构性问题是不一样的，而且都有很大的代表性，也就都可能成为互联网架构的热点。我们分头来看。

第三世界，发展中公司：这对应很多的新兴创业公司。这类公司发展很快同时面临的外部竞争压力也很大，他们的首要任务是将产品做好。所以其架构性问题是如何设计一个良好的有一定扩展能力并适合本产品的架构以支持产品的快速开发。在这个领域，大批的开源软件被使用。探讨新兴的开源软件和编程语言特性以及前端技术是这个领域的热点，一切能使开发变得简单的架构、方法都会直接被使用（像不像20世纪八九十年代的中国）。能够总结出一套LAMP这样经典的体系架构或者RESTful这样的架构风格，可能不是这类公司的任务，但却会给这些公司带来很大的指引作用。

第二世界，发达公司（这个名字有点儿怪）：这有点像我们现在国内的一些大公司。这些公司已经经历了很多年的发展，产品已经在一定程度上趋于稳定，但早期快速发展造成的基础不扎实带来的影响也渐渐体现出来。原先简单的产品架构变得日益复杂，各产品之间的联系越来越多，参与项目的人数越来越多，总体效率越来越低。这些公司要做的是将现有的系统进行整合，将共用的逻辑独立服务，推行各种标准化，推行各种管理上的流程化。SOA也算是这个领域一个比较泛的热点。这些公司正在经历着成功带来的痛苦（像不像有点儿颓的欧洲）。

第一世界，超级公司：这就是全球几大互联网巨头了。这些公司已经成功度过了第二阶段的痛苦，建立了良好的体系架构，现在正在享受良好的基础架构带来的巨大便利。这些公司面临的架构性问题，已经不能单纯用互联网架构来描述了，这需要整个计算机领域非常强大的体系架构进行支撑。超大规模的集群管理、数据存储和计算系统，作为整个体系架

邓毅，网易有道技术总监

构的基石。为大家极度推崇的Google所谓三架马车就是其中的代表作。这部分可能是大家最津津乐道的互联网热点，但其实真正需要如此大规模系统的公司，能真正接触到这些系统的人，都是少之又少。

栾义来：热点包括互联网数据的大规模处理和存储以及海量信息的实时检索。随着互联网应用的持续爆发，包括移动互联网和未来物联网的发展，这些都是加速度的发展，我们越来越面临大规模数据处理的问题。在海量数据里面人们也越来越希望可以实时地获得自己真正需要的那部分数据，这不是简单的索引问题，而是包括了规则定义的逻辑事件处理功能等，例如各大搜索引擎都在推出的实时搜索。

互联网架构下一步的发展趋势有哪些？

邓毅：第一是快速更新，第二是支持多媒体/视觉等数据。

冯大辉：现有的互联网架构向移动互联网迁移的过程中带来的技术挑战。新的技术（比如HTML5）对基础应用的改变带来的各种可能性。

侯震宇：现在是互联网大发展阶段，电子商务、移动互联网、SNS、搜索引擎无一不是炙手可热的大领域，全球互联网无论从市值还是流量影响力来看的几大巨头基本都集中在这几个领域，Web的架构性问题也主要在这几类产品中出现。太多产品特性我不想多谈，我想说的是这几年来互联网上的数据和流量的增长确实是太快了，这给很多大公司都带来了规模上的架构问题。这些问题再过几年会被现在的一些小公司同样经历。所以规模带来的问题将是互联网架构近期和长期都面临的一大问题。这个问题怎么解决？一个被炒烂了的概念就是云计算。

在做一个系统设计时需要具体情况具体分析。流量大的网站未必数据量大，比如Twitter的数据量其实很小。数据量大的系统也未必很复杂，比如Facebook的图片存储系统。OSDI'10的论文显示Facebook的图片数据有20PB，但其设计的Haystack却是个非常优雅而简单的系统。功能“简单”的系统未必设计实现起来就简单，比如Amazon Webservice里面大部分的服务都带有“Simple”的前缀，但仔细研究会发现其实每个设计都不简单。

综合来看，如何更好地利用多核进行编程、如何实现大规模的分布式存储、如何实现资源和计算的调度、如何使用虚拟机技术、如何实现大规模集群的自动管理、如何更好地定制或选型服务器、如何建设好网络和IDC，都将是巨头公司们需要考虑的问题。当然这一切系统的建设虽然有一些论文参考，但都必须根据自己公司业务的特点重新考虑，照搬任何一个已有的设计都是不靠谱的。国内的几家巨头都应该走类似的路，中小公司则没必要在这方面烧钱。

栾义来：有两个趋势，第一，从云端来讲，分布式计算将变成基本的编程框架，大量云设施的建设将大大简化编程的复杂性；第二，从通信协议上讲，HTTP协议所暴露的各种问题在一段时间内会被新的协议或其他方案所解决。

崔金峰：云计算相关技术的发展和技术的开源，各大公司采用的相关技术趋同，当然细节可能不同。互联网的架构会变得更简单、更稳健。

要成为一名优秀的互联网架构师，应该具备哪些条件？请给他们一些建议。

冯大辉：除了技术实现能力之外，如果要达到优秀的程度，还应该要比产品设计者更熟悉互联网产品，比运营更熟悉业务。或许这样有些求全责备了。

侯震宇：要想成为架构师，首先要具备足够扎实的计算机基础理论知识。万变不离其宗，即便是Google异常庞大的系统也仍然可以看作是冯·诺依曼结构。把GFS、BigTable当作硬盘内存，把MapReduce当作CPU计算，再加上数据来源做输入，用户服务做输出，如果再把IDC建设以及其电源制冷系统作为机箱风扇，整个一台计算机啊。GFS、Chubby等都是分布式系统几十年研究的产物，现在时髦的Erlang语言也是问世于二十多年前。当互联网服务由于产品规模的发展、硬件的更新换代等带来的新问题，其解决方法在现有互联网技术中找不到参照的时候，往往都要重新回到更基础的Computer Science领域去寻找答案，强大的基础知识在这时非

冯大辉，丁香园网站CTO

侯震宇，百度基础平台部架构师

常重要。

其次，要充分了解自己公司的产品，从自身产品特性出发进行架构分析和设计。架构师必然隶属于某个公司，为这个公司的某条或者某几条产品线服务。所以架构师必须了解自己产品的发展阶段，了解自己产品需要使用哪些技术。Google的架构纵然赏心悦目，却对绝大部分公司不适用。对于一些小公司来说，不用说去设计，即便是使用和研究都是意义不大的。任何架构设计都是依赖于产品特性的。学习大师级的设计首先要从学习他们的设计思路和初衷开始而不是设计本身。同样都是存储系统，Yahoo的PNUTS和Amazon的Dynamo就是完全不同的设计，这主要是因为他们面临的产品特性是完全不一样的，但就两个设计本身无所谓好坏，都是神作。

再次，要有足够的分析能力和经验。这些能力需要在工作中不断的积累和锻炼。架构师面临的问题往往都很大而且复杂，我们需要有足够的能力将其分解成若干个小问题或者是若干个曾经解决过的问题，并能抽象出一些问题的本质着力解决。我的一位大牛同事曾经说过，做设计要提前做back-of-the-envelop calculation，不要只知道试错。我深以为然，比如在系统设计之初就应该知道系统可能的瓶颈在哪里，是硬盘的IOPS还是吞吐，是网卡还是总线带宽，抑或是系统中断次数，都可能成为系统的瓶颈。我们需要根据系统的服务需求，推算出系统的负荷并根据硬件特性设计出适合的架构。

最后，有足够广的视野和能够深入做下去的决心。架构师是一个杂家，什么东西都应该懂一些。对自己所处领域的其他人的做法要有所了解并加以分析，这样当我们遇到类似问题时就会得心应手。当然万事不能只说只看，关键也要实际去做。做过一个系统与看过和用过一个系统的理解是完全不一样的。架构师虽然工作相对比较忙，但是要有足够的决心能够深入到需要深入的领域去。

栾义来：我的建议包括三个方面：第一，应争取能进入大公司工作，只有实际解决过海量应用问题才能真正掌握问题的核心；第二，应多思考互联网应用所反映的现实世界问题，避免陷入纯技术论的陷阱；第三，简单就是美、少即多，具备美的理解和一定的哲学思考，有助于迅速解决问题。

崔金峰：第一，基础还是最重要的，比如通信、内存、线程、文件、算法、相关分布技术（分布式存储、分布式计算、分布式数据库等）这些都是必须精通的；第二，经验很重要，能在不同的场景中，使用不同的技术，并且力求简单和降低成本；第三，设计系统架构的时候，先关注瓶颈问题，瓶颈到底出在哪儿？能够从大的方面及小的关键细节，将未来长远扩展等都能描述得非常清楚。

邓毅：首先要有丰富的程序、系统设计经验；其次要学会了解技术需求乃至用户需求；最后要热爱新东西、有创新精神。

在大家心目中，最理想的互联网架构是什么？

侯震宇：没有什么理想的架构能够包打天下，实际情况下都要根据不同的需求去做不同的设计。就服务设计风格来说，我喜欢RESTful的设计风格。就产品架构来说，我个人比较喜欢Amazon的架构，这对于众多中小网站甚至是一些大型网站带来的益处太大了，真的希望AWS哪天进入中国或者中国自己有类似的服务。

栾义来：不同应用系统的要求是不一样的，架构没有好坏之分，只有合适与否。如果能有成熟的基础互联网协议和基础设施可以将应用分发、事务协调、节点自治等核心问题和谐并且开放地管理好，会是非常好的事情，目前的解决方案太笨重并且私有了。

崔金峰：还是以Google为核心的架构思想，Google的相关思想和技术这些年来推动了互联网技术的发展。

栾义来，凡客诚品架构总监

邓毅：理想的架构就是能把解决实际问题、系统的复杂度、系统的美感很好地平衡的架构。

冯大辉：能够适时地支撑业务的发展。不过度超前，也不可过于落后。

Sina App Engine架构

——云计算时代的分布式Web服务解决方案

■ 文 / 丛磊

Sina App Engine（简称SAE）是新浪研发中心于2009年上半年开始内部开发，并在2009年11月3日正式推出第一个Alpha版本的国内首个公有云计算平台（http://sae.sina.com.cn），是新浪云计算（简称浪云）战略的核心组成部分。

SAE作为国内的公有云计算，借鉴吸纳了Google、Amazon等国外公司的公有云计算的成功技术经验，并很快推出具有自身特色的云计算平台。SAE选择PHP作为首选的支持语言，Web开发者可以在Linux/Mac/Windows上通过SDK或者Web版在线SDK进行开发、部署、调试，团队开发时还可以进行成员协作，不同的角色将对代码、项目拥有不同的权限。SAE还提供了一系列分布式计算、存储服务供开发者使用，包括分布式文件存储、分布式数据库集群、分布式缓存、分布式定时服务等，这些服务将大大降低开发者的开发成本。同时又由于SAE整体架构的高可靠性和新浪的品牌保证，大大降低了开发者的运营风险。另外，作为典型的云计算，SAE采用“所付即所用，所付仅所用”的计费理念，通过日志和统计中心精确的计算每个应用的资源消耗（包括CPU、内存、磁盘等）。

总之，SAE就是分布式Web服务的开发、运行平台。

SAE的目标和发展

云计算在国外已经有4~5年的历史。2006年，Amazon就推出了以EC2为代表的公有云计算，并且实现了大规模盈利；2008年，Google推出了以Google App Engine为代表的公有云计算。国内的云计算一直是炒得很厉害，各大互联网公司都在宣传，但真正有技术实力做出来而又对外公开使用的少之又少。

从2004年开始，新浪就开始了私有云方向的研究和实践，以此为基础的动态应用平台目前已经支撑新浪内部的绝大部分业务。从2008年起，新浪又启动了“浪云”的公有云计算计划，相继开发了分布式队列服务、P2P文件系统、分布式计算框架等一系列基础服务。实际SAE就是“浪云”战略的产物。

SAE从架构设计和代码编写开始，就明确了自身的两个目标：第一，做公有云计算平台，公有云不同于私有云，更强调安全性和可靠性，这也对整体的架构设计和技术实现提出了更苛刻的要求；第二，为分布

表1　SAE和传统的虚拟主机托管VPS的主要区别

类项	SAE	VPS
核心用户	Web开发者	无核心用户
使用方式	服务使用	设备租用
目标	力争覆盖Web服务所有需求，提供多种服务给开发者使用	仅基本需求
SLA（服务承诺）	高可靠性及严格的服务承诺	依服务商变化，无严格协议
计费方式	所付即所用，所付仅所用	预付费，无精确计费

作者简介：
丛磊，资深架构师，SAE技术主管，擅长应用算法、编译器实现、分布式系统设计、人工智能博弈论等，2006年毕业后加入新浪，2008年开始带领技术团队从事云计算方向的研究与开发。

式Web服务提供一整套的解决方案，SAE争取提供开发者开发Web应用过程中所用到的所有服务。

经过技术团队一年的开发，SAE目前已经提供了十多种服务，整体上分为计算型和存储型，计算型又包括同步计算和异步计算，而存储型则分为持久化存储和非持久化存储，如表2所示。

表2　SAE提供的服务

服务名称	类型	说明
HTTP+PHP	同步计算	带SAE沙盒的Apache和Zend为用户提供Web计算服务
Stor	持久化存储	提供分布式文件存储
Memcache	非持久化存储	提供分布式缓存服务
RDC	持久化存储	分布式数据库集群，提供MySQL服务
Taskqueue	异步计算	异步离线轻量级任务队列，HTTP方式调用
DeferredJob	异步计算	异步离线重量级任务队列，系统方式调用
Cron	异步计算	分布式定时服务
FetchURL	同步计算	分布式抓取服务
Tmpfs	非持久化存储	提供临时文件存储
Appconfig		提供应用配置功能，取代Apache htaccess
Mail	异步计算	邮件发送服务
Image	同步计算	图像处理服务
XHProf	同步计算	Facebook提供的强大的PHP调优工具
其他工具		
SDK		Windows GUI SDK、Linux/Mac command line SDK
Online SDK		在线代码编辑器

SAE于2009年11月3日发布了Alpha1版本，2010年2月1日发布了Alpha2版本，2010年9月1日发布了Beta版本，经过将近一年的不断完善和改进，尽管SAE一直没有开放注册（实际云计算的模式也不以注册用户的规模为评价标准），但已经拥有了一批有价值的App和粉丝开发者。截止10月1日，SAE拥有开发者4000多名，App总数3000个，活跃App将近1000个，每天独立代码部署行为超过1000次。

整体架构

SAE从架构上采用分层设计，从上往下分别为反向代理层、路由逻辑层、Web计算服务池。而从Web计算服务层延伸出SAE附属的分布式计算型服务和分布式存储型服务，具体又分成同步计算型服务、异步计算型服务、持久化存储服务、非持久化存储服务。各种服务统一向日志和统计中心汇报如图1所示。

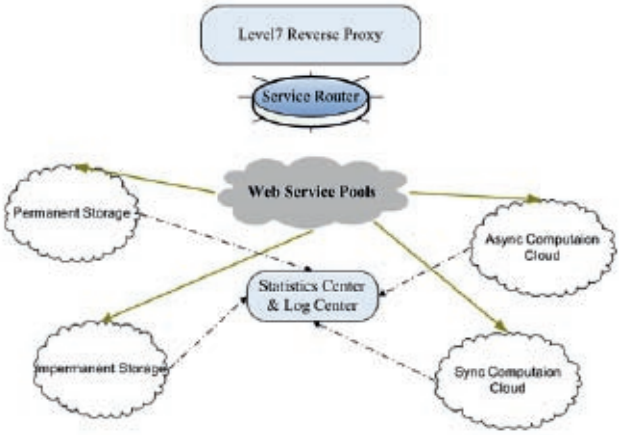


图1　SAE整体架构图

7层反向代理层：HTTP反向代理，在最外层，负责响应用户的HTTP请求、分析请求并转发到后端的Web服务池上，提供负载均衡、健康检查等功能。

服务路由层：逻辑层，负责根据请求的唯一标识，快速地映射（O(1)时间复杂度）到相应的Web服务池及相应的硬件路径。如果发现映射关系不存在或者错误，则给出相应的错误提示。该层对用户隐藏了很多具体地址信息，使开发者无须关心服务的内部实际分配情况。

Web服务池：由一些不同特性的Web服务池组成。每个Web服务池实际是由一组Apache Server组成的，这些池按照不同的SLA提供不同级别的服务。每个Web服务进程实际处理用户的HTTP请求，进程运行在HTTP服务沙盒内，同时还同样内嵌运行在SAE沙盒内的PHP解析引擎。用户的代码最终通过接口调用各种服务。

日志和统计中心：负责对用户所使用的所有服务的配额进行统计和资源计费，这里的配额有两种，一种是分钟配额，用来保证整个平台的稳定；一种是天配额，用户可以给自己设定每天资源消耗的最高上限。日志中心负责将用户所有服务的日志

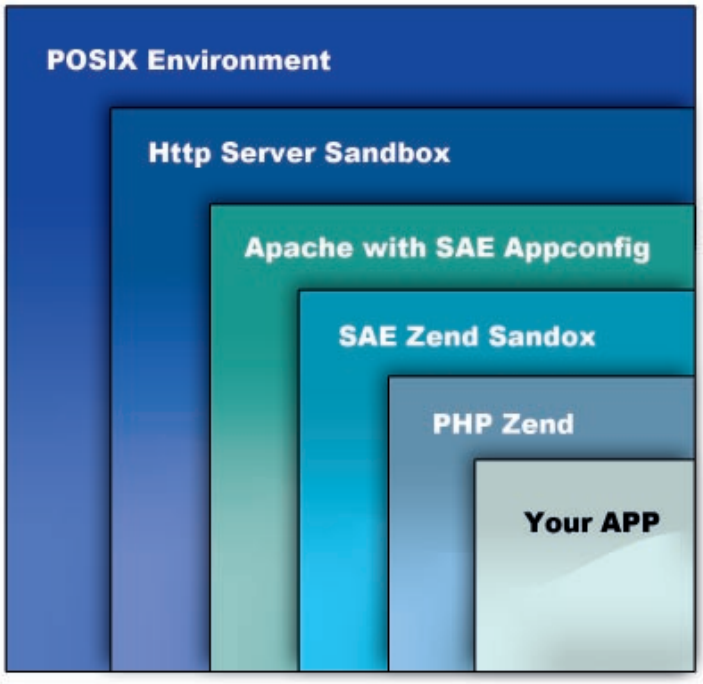


图2　SAE沙盒结构图

汇总并备份，并提供检索查询服务。

各种分布式服务：SAE提供Web应用开发所需的大多数服务，同时支持接入第三方服务，用户可以通过StdLib（可以理解为SAE PHP版的STL）很方便地调用它们。

真正的用户代码是跑在SAE提供的Web运行环境下的，为了提供公有云计算特有的安全性，SAE设计多层沙盒来保证用户应用之间的隔离性，如图2所示。

最内层的就是用户代码，大部分PHP代码不需要做任何修改就可以跑在SAE平台上，小部分代码需要做一些修改以适应SAE的平台特性。这主要有两部分：第一，SAE因为安全性禁用了本地I/O，所以fwrite等函数需要修改为使用Tmpfs读写本地临时文件或者直接通过Stor读写我们的分布式文件存储；第二，用户在SAE上不能通过Curl访问非“SAE域”的资源，用户有抓取公网资源等需求，需要修改为调用FetchURL服务。我们提供了PHP Wrapper以方便用户的修改。

PHP Zend为标准的PHP官方解释器，我们采用的版本为5.3.0。

SAE Zend Sandbox为一个逻辑概念，为用户的代码运行提供良好的隔离性。这里有两个层面，第一是通过标准的php.ini，我们设定了一些特殊配置和禁用函数；第二，为了达到一些php.ini无法实现的沙盒功能，我们对Zend解释器核做了一些改进，以便通过用户标识将资源进行隔离。另外我们还把一些SAE的特定服务也在Zend层做了融合。

Apache为标准的Apache Web Server，版本为2.2。不过我们禁用了htaccess，并提供了自己实现的替换方案AppConfig。用户可以通过类自然语言的方式编写AppConfig，如 - compress: if(out_header[“Content-Length”] >= 500) compress 表示按条件启动页面压缩。目前AppConfig提供的功能有：目录默认页面、自定义错误页面、压缩、页面重定向、页面过期、设置响应头的content-type、设置页面访问权限。我们选择自行实现AppConfig还有一个考虑，就是因为传统Apache的htaccess因为要按目录递归方式合并配置文件，效率不能满足SAE的需求。

HTTP Server沙盒为Apache的安全可靠运行提供了多种保护功能，比如防止某个用户恶意占用连接数从而导致整个Web服务不正常。

最外层的是标准POSIX环境，目前我们的服务跑在Linux 2.6上。

上面就是对SAE整体架构的概述，接着将详细讨论我们在架构设计上的一些具体考量点。

扩展性

扩展性是分布式系统的两个主要目的之一，SAE作为公有云计算，同样把服务的扩展性作为架构设计的重要指标，要求在用户增长、压力提升的情况下，可以实现自动的服务扩展，同样当压力降低时，可以将服务收缩，以节约资源，整个过程无须人工参与。SAE人工只需做好容量规划和管理。目前国外的公有云计算架构的扩展性主要有静态和动态两个思路。

静态扩展：用户和资源有强绑定关系。最典型的例子为Amazon的EC2和Ruby云计算平台Heroku，用户申请的资源和用户有严格的一对一关

系，换句话说，A用户申请的虚拟机在A退还资源前，B用户不能使用，哪怕A用户的虚拟机处于闲置状态。

动态扩展：用户和资源没有强绑定关系。最典型的例子为Google App Engine，用户申请的资源和用户没有严格的一对一关系，换句话说，处理A用户请求的进程在处理完之后，可以马上处理B用户的请求。

两种扩展性各有利弊，静态扩展的长处是为平台提供了良好的隔离性，资源可以固定映射在某个用户下，但缺点是资源利用率不高；动态扩展的长处是资源利用率高，这样整个云计算平台的成本会很低，但缺点是对隔离性有更高的要求，因为资源可以在很短的时间被多个用户使用。相比较，在安全性上，动态扩展要比静态扩展的技术门槛更高。

在SAE平台上，我们采用以动态扩展为主、静态扩展为辅的兼而有之的设计。在Web计算池层是典型的动态扩展。而在SAE的某些服务中，又是以静态扩展的方式展现，如RDC（Relational DB Cluster）分布式数据库集群，当用户申请了MySQL服务，我们就会在RDC后端根据SLA创建一主多从的DB给用户，在用户显式删除该DB前，该DB都不会被别人使用。当然，通过RDC，任何一个用户也无需知道后端DB的实际地址，只需访问RDC统一的Host和Port即可。

高可靠性（High Availability，简称HA）

HA是分布式系统的另一个主要目的，SAE同样以提供服务的高可靠性为架构设计的重要指标。HA的实现途径主要有两个：一个是硬件保证，另一个是架构的冗余设计。

在SAE平台上，所有服务器都是新浪标准采购的硬件设备，运行在国内最好的机房内，网络资源方面则享用门户网站所使用的带宽环境。另外，所有的硬件设备都有专门的运维部门负责，故障的响应速度和新浪内部服务一样。

在架构设计上，SAE通过对所有服务都进行冗余设计来提供服务的高可靠性。这里的服务可以分成计算型和数据型两种类别讨论。

针对计算型服务，冗余设计就是程序在多节点运行。我们要求SAE所有的内部代码程序要做到Stateless（无状态依赖），即无依赖部署无依赖启动，随时终止进程随时重启进程，这样一旦出现机器故障或者程序自身Bug时，所有进程能够随着硬件环境的重新恢复而在第一时间重启。而多点执行的程序可以保证，当某些程序出现故障时，整个系统仍然能够正常提供服务。

计算型程序多点部署，会带来一致性问题，最主要的困扰就是选举问题，如何在多个节点中选出一个主节点来执行。比如SAE上的分布式定时服务Cron，采用多点部署方式，多个计算节点相互隔离，通过时钟同步服务同时触发用户设定的定时任务，但要求只能有一个节点负责执行。为了解决这个问题，SAE设计出了一套分布式锁算法来提供选举服务。该算法可以在牺牲某些特定条件下的一致性来提供比Paxos算法更高的可靠性（3台机器在最高任意2台机器发生故障的情况下整个选举过程仍然正常，而Paxos算法最多容忍1台）。目前，该算法正在申请专利，并广泛应用在SAE内部。

针对数据型服务，SAE主要是通过复制来保证服务的高可靠性。SAE上的数据存储服务普遍采用被动复制和主动复制两种方式。如SAE上MySQL之间的主从Binlog同步就是典型的被动复制，用户只写写库，数据从写库同步到多个读库中。Taskqueue、DeferredJob等服务也采用被动复制的方式，用户的任务描述会写到主内存级队列中，主队列利用后台线程将写操作同步到从队列上，一旦主队列发生故障，从队列会快速的切换为主队列。另外SAE上也有部分服务采用主动复制（双写复制）的方式来保证HA，比如Cron，当用户通过App的工程配置文件appconfig.yaml设定定时任务时，任务信息会以多写的方式写到多个持久化DB中，以供后续的事件触发。

另外，SAE在整体架构设计时，充分考虑服务之间的“优雅降级”，尽量降低服务之间的耦合度，我们要求任何一个服务都不要假设其他服务是可靠的。目前在SAE平台上的所有服务均不存在单点设计，服务的平均HA在99.9%，即年平均服务不可用时间在8~9个小时之间。

未来

未来SAE有两个重要工作，一个是为开发者提供公平完善的评估奖励机制，并开放多种支付接口，目的就是早日形成和开发者之间双赢的良性循环，来吸引更多的开发者在SAE开发应用；另一个就是在保证现有服务稳定的前提下，为用户提供更多、更便捷的服务，涉及CDN、网络代理、Key-Value数据库等。以后我会继续给大家介绍SAE内部具体服务的架构和技术实现细节。

分布式文件系统FastDFS架构剖析

■ 文 / 余庆

FastDFS是一款类Google FS的开源分布式文件系统，它用纯C语言实现，支持Linux、FreeBSD、AIX等UNIX系统。它只能通过专有API对文件进行存取访问，不支持POSIX接口方式，不能mount使用。准确地讲，Google FS以及FastDFS、mogileFS、HDFS、TFS等类Google FS都不是系统级的分布式文件系统，而是应用级的分布式文件存储服务。

FastDFS的设计理念

FastDFS是为互联网应用量身定做的分布式文件系统，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标。和现有的类Google FS分布式文件系统相比，FastDFS的架构和设计理念有其独到之处，主要体现在轻量级、分组方式和对等结构三个方面。

轻量级

FastDFS只有两个角色：Tracker server和Storage server。Tracker server作为中心结点，其主要作用是负载均衡和调度。Tracker server在内存中记录分组和Storage server的状态等信息，不记录文件索引信息，占用的内存量很少。另外，客户端（应用）和Storage server访问Tracker server时，Tracker server扫描内存中的分组和Storage server信息，然后给出应答。由此可以看出Tracker server非常轻量化，不会成为系统瓶颈。

FastDFS中的Storage server在其他文件系统中通常称作Trunk server或Data server。Storage server直接利用OS的文件系统存储文件。FastDFS不会对文件进行分块存储，客户端上传的文件和Storage server上的文件一一对应。

众所周知，大多数网站都需要存储用户上传的文件，如图片、视频、电子文档等。出于降低带宽和存储成本的考虑，网站通常都会限制用户上传的文件大小，例如图片文件不能超过5MB、视频文件不能超过100MB等。我认为，对于互联网应用，文件分块存储没有多大的必要。它既没有带来多大的好处，又增加了系统的复杂性。FastDFS不对文件进行分块存储，与支持文件分块存储的DFS相比，更加简洁高效，并且完全能满足绝大多数互联网应用的实际需要。

在FastDFS中，客户端上传文件时，文件ID不是由客户端指定，而是由Storage server生成后返回给客户端的。文件ID中包含了组名、文件相对路径和文件名，Storage server可以根据文件ID直接定位到文件。因此FastDFS集群中根本不需要存储文件索引信息，这是FastDFS比较轻量级的一个例证。而其他文件系统则需要存储文件索引信息，这样的角色通常称作NameServer。其中mogileFS采用MySQL数据库来存储文件索引以及系统相关的信息，其局限性显而易见，MySQL将成为整个系统的瓶颈。

FastDFS轻量级的另外一个体现是代码量较小。最新的V2.0包括了C客户端API、FastDHT客户端API和PHP extension等，代码行数不到5.2万行。

分组方式

类Google FS都支持文件冗余备份，例如Google FS、TFS的备份数是3。一个文件存储到哪几个存储结点，通常采用动态分配的方式。采用这种方式，一个文件存储到的结点是不确定的。举例说明，文件备份数是3，集群中有A、B、C、D四个存储结点。文件1可能存储在A、B、C三个结点，文件2可能存储在B、C、D三个结点，文件3可能存储在A、B、D三个结点。

FastDFS采用了分组存储方式。集群由一个或多个组构成，集群存储总容量为集群中所有组的存储容量之和。一个组由一台或多台存储服务器组成，同组内的多台Storage server之间是互备关系，同组存储服务器上的文件是完全一致的。文件上传、下载、删除等操作可以在组内任意一台Storage server上进行。类似木桶短板效应，

作者简介：

余庆，现在淘宝网Java中间件团队从事Java基础平台研发工作，有10年互联网开发和架构经历，曾担任新浪网开发工程师、雅虎中国架构师。开源分布式文件系统FastDFS和分布式哈希系统FastDHT的作者，对分布式数据存储架构有比较深入的研究。

一个组的存储容量为该组内存储服务器容量最小的那个，由此可见组内存储服务器的软硬件配置最好是一致的。

采用分组存储方式的好处是灵活、可控性较强。比如上传文件时，可以由客户端直接指定上传到的组。一个分组的存储服务器访问压力较大时，可以在该组增加存储服务器来扩充服务能力（纵向扩容）。当系统容量不足时，可以增加组来扩充存储容量（横向扩容）。采用这样的分组存储方式，可以使用FastDFS对文件进行管理，使用主流的Web server如Apache、nginx等进行文件下载。

对等结构

FastDFS集群中的Tracker server也可以有多台，Tracker server和Storage server均不存在单点问题。Tracker server之间是对等关系，组内的Storage server之间也是对等关系。传统的Master-Slave结构中的Master是单点，写操作仅针对Master。如果Master失效，需要将Slave提升为Master，实现逻辑会比较复杂。和Master-Slave结构相比，对等结构中所有结点的地位是相同的，每个结点都是Master，不存在单点问题。

FastDFS的架构

图1展示的是FastDFS的系统架构。

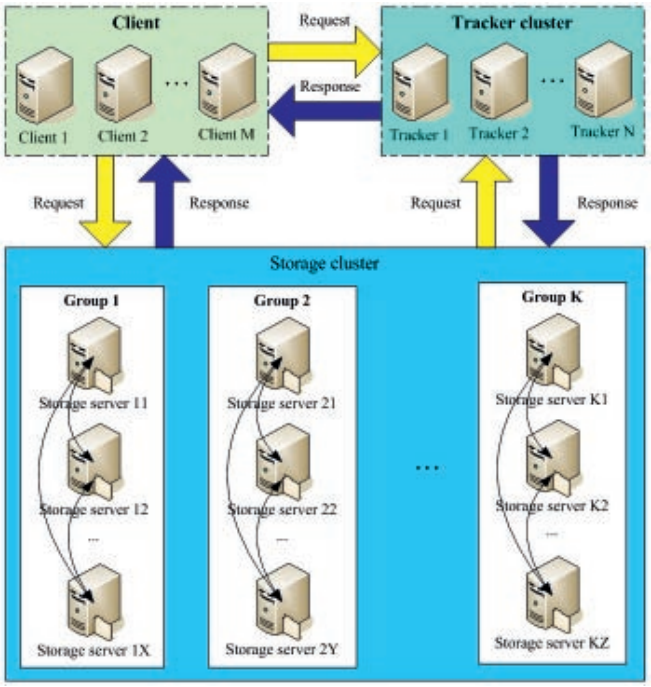


图1　FastDFS的系统架构

从图1可以看出，Tracker server之间相互独立，不存在直接联系。

客户端和Storage server主动连接Tracker server。Storage server主动向Tracker server报告其状态信息，包括磁盘剩余空间、文件同步状况、文件上传下载次数等统计信息。Storage server会连接集群中所有的Tracker server，向他们报告自己的状态。Storage server启动一个单独的线程来完成对一台Tracker server的连接和定时报告。需要说明的是，一个组包含的Storage server不是通过配置文件设定的，而是通过Tracker server获取到的。

不同组的Storage server之间不会相互通信，同组内的Storage server之间会相互连接进行文件同步。

Storage server采用binlog文件记录文件上传、删除等更新操作。binlog中只记录文件名，不记录文件内容。

文件同步只在同组内的Storage server之间进行，采用push方式，即源头服务器同步给目标服务器。只有源头数据才需要同步，备份数据并不需要再次同步，否则就构成环路了。有个例外，就是新增加一台Storage server时，由已有的一台Storage server将已有的所有数据（包括源头数据和备份数据）同步给该新增服务器。

Storage server中由专门的线程根据binlog进行文件同步。为了最大程度地避免相互影响以及出于系统简洁性考虑，Storage server对组内除自己以外的每台服务器都会启动一个线程来进行文件同步。

文件同步采用增量同步方式，系统记录已同步的位置（binlog文件偏移量）到标识文件中。标识文件名格式：{dest storage IP}_{port}.mark，例如：192.168.1.14_23000.mark。

文件上传和下载的交互过程

接下来我们一起看一下文件上传和下载的交互过程。文件上传和下载流程分别如图2、图3所示。文件上传流程的步骤如下：

1. Client询问Tracker server上传到的Storage server；

2. Tracker server返回一台可用的Storage server，返回的数据为该Storage server的IP地址和端口；

3. Client直接和该Storage server建立连接，进行文件上传，Storage server返回新生成的文件ID，文件上传结束。

文件下载流程的步骤如下：

1. Client询问Tracker server可以下载指定文件的Storage server，参数为文件ID（包含组名和文件名）；

2. Tracker server返回一台可用的Storage server；

3. Client直接和该Storage server建立连接，完成文件下载。

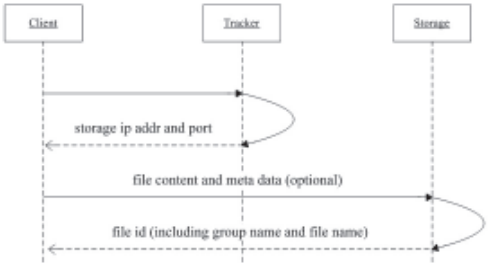


图2　文件上传流程

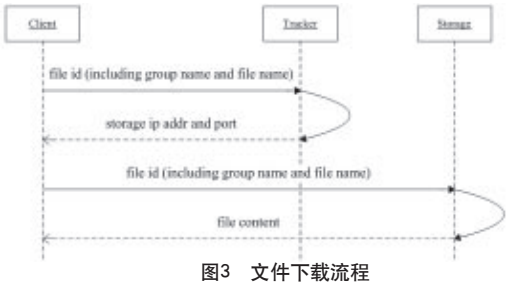


图3　文件下载流程

文件同步延迟问题的提出

客户端将一个文件上传到一台Storage server后，文件上传工作就结束了。由该Storage server根据binlog中的上传记录将这个文件同步到同组的其他Storage server。这样的文件同步方式是异步方式，异步方式带来了文件同步延迟的问题。新上传文件后，在尚未被同步过去的Storage server上访问该文件，会出现找不到文件的现象。FastDFS是如何解决文件同步延迟这个问题的呢？

文件的访问分为两种情况：文件更新和文件下载。文件更新包括设置文件附加属性和删除文件。文件的附加属性包括文件大小、图片宽度、图片高度等。FastDFS中，文件更新操作都会优先选择源Storage server，也就是该文件被上传到的那台Storage server。这样的做法不仅避免了文件同步延迟的问题，而且有效地避免了在多台Storage server上更新同一文件可能引起的时序错乱的问题。

那么文件下载是如何解决文件同步延迟这个问题的呢？

要回答这个问题，需要先了解文件名中包含了什么样的信息。Storage server生成的文件名中，包含了源Storage server的IP地址和文件创建时间等字段。文件创建时间为UNIX时间戳，后面称为文件时间戳。从文件名或文件ID中，可以反解出这两个字段。

然后我们再来看一下，Tracker server是如何准确地知道一个文件已被同步到一台Storage server上的。前面已经讲过，文件同步采用主动推送的方式。另外，每台storage server都会定时向tracker server报告它向同组的其他storage server同步到的文件时间戳。当tracker server收到一台storage server的文件同步报告后，它会依次找出该组内各个storage server（后称作为S）被同步到的文件时间戳最小值，作为S的一个属性记录到内存中。

FastDFS对文件同步延迟问题的解决方案

下面我们来看一下FastDFS采取的解决方法。

一个最简单的解决办法，和文件更新一样，优先选择源Storage server下载文件即可。这可以在Tracker server的配置文件中设置，对应的参数名为download_server。

另外一种选择Storage server的方法是轮流选择（round-robin）。当Client询问Tracker server有哪些Storage server可以下载指定文件时，Tracker server返回满足如下四个条件之一的Storage server：

- 该文件上传到的源Storage server，文件直接上传到该服务器上的；
- 文件创建时间戳 < Storage server被同步到的文件时间戳，这意味着当前文件已经被同步过来了；
- 文件创建时间戳=Storage server被同步到的文件时间戳，且（当前时间—文件创建时间戳） > 一个文件同步完成需要的最大时间（如5分钟）；
- （当前时间—文件创建时间戳） > 文件同步延迟阈值，比如我们把阈值设置为1天，表示文件同步在一天内肯定可以完成。

结束语

看了上面的介绍，你是否认为FastDFS比较简洁高效呢？原雅虎同事——一位比较资深的系统架构师听完FastDFS介绍后，作出这样的评价：“FastDFS是穷人的解决方案”。他的意思是说FastDFS把简洁和高效做到了极致，非常节约资源，中小网站完全用得起，这是对FastDFS的极大认可和褒奖。

FastDFS从2008年7月发布至今，已推出31个版本，后续完善和优化工作正在持续进行中。目前已有多家公司在生产环境中使用FastDFS，相信通过我们的不懈努力，FastDFS一定会越来越好！

从#NewTwitter新界面说起

■ 文 / 邝宇恒

作者简介：

邝宇恒，腾讯广州研发中心高级工程师，负责QQMail基础架构研发，持续提高系统性能和可靠性。兴趣广泛，热衷研究新鲜事物。热爱折腾，热爱分享，推崇简单、踏实、开放的学习和工作风格。最近在帮忙折腾自由交流的Barcamp Guangzhou Unconference。

不久前，Twitter开始了其Web网站有史以来最大规模的一次改版，在激动人心的体验改进背后，我们还可以注意到它在网站架构和技术上的变革。Twitter开发团队广受关注的文章《The Tech Behind the New Twitter.com》描述了这样的变化。

文章提到了网站架构的两点重要变化。首先，新Twitter.com网站成为一个普通的API客户端，与其他第三方Twitter应用一样，调用标准的Twitter API接口；其次，新Twitter.com网站前端以JavaScript作为核心技术，以客户端JavaScript作为展现和控制用户界面的主要手段。以用户首页为例，旧版Twitter页面主体是消息内容的HTML，而新版已是一个与Twitter API接口输出结果相同的JSON数组，再由客户端JavaScript渲染页面展现；异步更新消息时，旧版Twitter使用了特殊的接口，获取更新消息的HTML片段，而新版则是直接从标准的API接口api.twitter.com抓取。从这个典型页面即可看出这些变化的端倪。

NewTwitter的思考

这样的架构改变，背后的原因更发人深思。

Twitter从创建开始就是一个与移动互联网深度结合的应用，移动互联网的快速增长，使这样的结合更加紧密，很大一部分用户在移动设备上，甚至主要在移动设备上使用Twitter。从智能机到平板，不同设备有不同的特性和体验习惯，单一版本的mobile.twitter.com显然无法满足需求，各种设备上都需要有特别定制的客户端版本——不管是基于本地客户端还是Web技术；而在桌面上，随着Twitter深入生活的方方面面，用户需求也变得多种多样——功能更丰富的界面、与其他社交平台整合、轻量级浏览器插件，早已远不是一个官方Web界面能满足的。

满足如此多样化需求的是极为丰富的第三方客户端，支持这些客户端的是强大的Twitter API。面对如此众多的设备，以及如此丰富多样的需求，官方Web界面的重要性在降低，而API的重要性则越来越高。在Twitter，这两者实现了逆转：不再为Web界面维护特殊架构，网站架构以API为核心，而Web界面仅是它的一个普通客户端。由此，更多的资源能投入到API的开发维护中，更好地为更多的客户端服务。

在Web界面开发方面，Twitter也面临了完全不同的环境。Twitter在2006年上线，正是Ajax大红大紫的时候，Twitter却依然使用了几乎完全静态的界面。我想，其中的重要原因在于浏览器能力问题，在IE6等过时浏览器上，很难在保证响应速度的前提下实现丰富的用户界面，还不如极尽简单。之后Twitter Web界面进行多次改进，仍不脱尽量减少浏览器端开销的原则。

四年之后，一切都不同了。Web开发者恨之入骨的IE6终于淡出市场。新的浏览器大战中的选手们，首先都具有数倍于前代的JavaScript执行效率，开发者可以更放心地在客户端使用JavaScript进行大量处理。新浏览器还在积极地支持新的标准，今天，我们已经可以放心地使用

LocalStorage等特性，因为它已成为各浏览器的默认功能，HTML5的更多新能力也会迅速普及，成为默认功能。更令人振奋的是iPhone、Android等智能手机系统的浏览器，也同样具有这样的能力。新的Twitter Web界面，正是充分利用这些新的浏览器能力的典型。

不仅是Twitter，所有的Web开发者都面临着这些变化。与Twitter一样，这些变化可能会深刻地影响Web应用的整体架构。我把这些变化总结为：

- 空前丰富的设备和使用需求，传统Web界面不再一统江湖甚至不占主要地位。支持开放API和多种前端是系统架构的必备能力，应在初始设计时就进行考虑。
- 浏览器能力与几年前不可同日而语。考虑前端技术时，可抛开以往的许多假设（如IE6兼容），充分利用新浏览器能力，作全新考虑。

面对这样的变化，我想会有许多应用的架构趋向新Twitter的模式。将业务逻辑封装成功能强大、风格一致、接口稳定且有充足安全机制的API服务，作为系统的核心；围绕API服务开发各种不同前端，并将API开放给第三方使用。

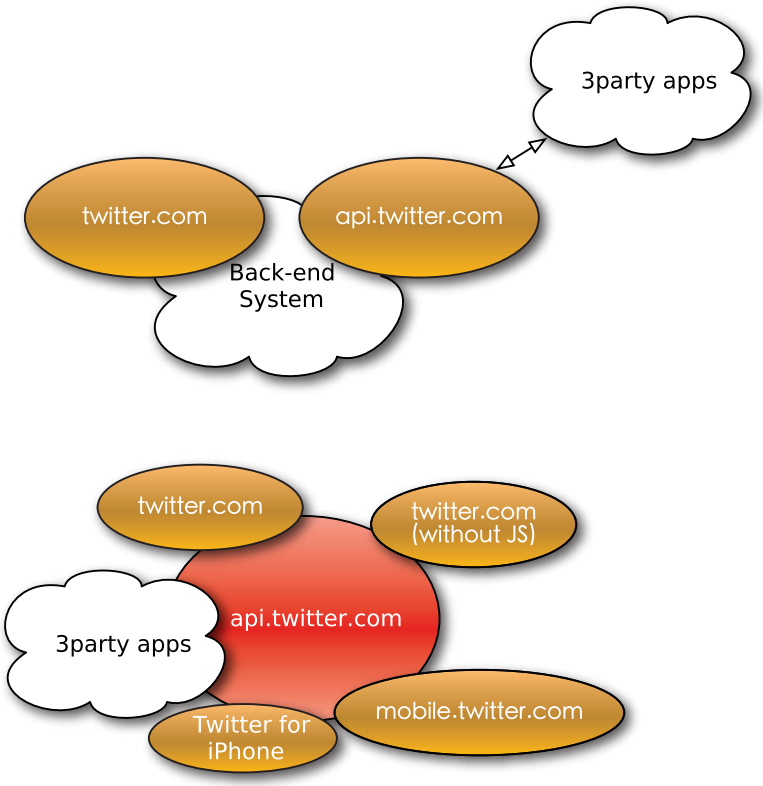


图1　Twitter架构改变示意图

从业务逻辑层到API服务

设计这样的API服务，第一步无疑是整理业务逻辑，保证业务逻辑与界面逻辑的分离，将业务逻辑封装成API才有可能。这两者的分离应是老生常谈，许多项目至少在设计图上显著划分了业务逻辑层和界面层，MVC等设计模式也在促进这样的分离。可实际情况并不这么乐观。

许多Web应用项目采用MVC结构，在这样的结构中，Model应包含全部业务逻辑，但实际情况并不如此。例如在社区应用中，发文章可以加积分，一个很常见的实现是：文章是一个Model，积分是另一个Model，在Controller中，分别调用这两个Model完成发文章和加积分功能！

这个设计的问题是显而易见的，若要写一套全新界面，它面临的是一个文章Model和一个可以任意操纵的积分Model，新界面实现的积分操作逻辑很有可能与旧界面不一致——或者根本忘了操作积分。这样的设计问题也许还跟Ruby on Rails等快速开发框架的普及有关，框架默认支持一个Model与一个数据库表的对应，这个支持如此好，以至于人们甚至忘了Model应对应于业务逻辑，而将其看作数据库操作封装，转而将其他逻辑一股脑儿扔到最方便、什么都能管的Controller里，与界面逻辑混在一起。

从快速开发Web应用角度看，这样做问题不大，无论代码放在哪里，能出页面就行。然而，当你严肃考虑应用必然有多个不同前端的现实时，就会发现这样的Ad-hoc方法再不可行，每个业务逻辑的变更都有考虑全部前端的支持或兼容，通用的业务逻辑与特殊的前端界面必须严格分离。设计原则、设计模式可能都是老生常谈，但新情况逼迫我们认真重新考虑。

保证业务逻辑与界面逻辑的分离不可能仅靠小心的设计。即使初始设计确实划分清晰、干净整洁，网站上线后，千奇百怪的需求汹涌

而来，让人手忙脚乱，哪有时间来设计规划？项目很快又会变成一坨稀粥的状态。保证它的，应该是统一的接口模式和强有力的设计约束和框架的支持。

接口模式

业务接口的设计——也就是API服务的接口设计，从未变得如此重要。在以往，它也许只是一个内部设计、代码组织的问题，今天，在以API服务为中心的架构下，它将实实在在地影响业务开发。一个可以提供给多种前端并开放给第三方使用的接口，必须满足下面的要求。

- **功能完善**：接口需要支持完整的前端应用。这不再是以往设计开放API的那种只开放少数接口的方法，整个系统的核心功能都需要通过API服务提供。
- **易于学习和使用**：API服务的使用者不再是单一的Web前端，内部和外部多个团队、多种应用都需要使用该API，这对其易用性提出了更高要求。API应该有统一的接口模式，方便使用者学习；API应该提供HTTP、XML等标准的交互方式，不必依赖特定客户端库，方便不同开发平台使用；API最好能够符合已有的通用标准，以获得最好的周边支持，降低迁移成本。
- **接口稳定**：作为开放给第三方的接口，只要开放出去，就已没有修改的机会；即使只限内部使用，只要有客户端产品使用该接口，接口也就几乎没有机会废弃或改变——这跟Web不同，总有用户不升级自己的客户端。

实现这些需求相当不易，Twitter API是一个很好的范例，可以给我们的应用API设计提供参考。

Twitter首先根据业务特征划分了若干资源，如Timeline（消息列表）和Users（用户），并为每种资源制定标准的表达方式。统一的表达是同种资源的重要特征：不论是public_timeline还是home_timeline，消息列表的结构都是相同的，其中每条推的内容结构又与通过statuses/show/:id的单条消息接口取得的结构相同，每条推包含的作者信息，又与通过users接口获得的相同……

下一步则是刻画对资源的操作，最重要的操作无疑是对资源的查询。例如，Timeline资源有user_timeline（订阅消息列表）、user_timeline（某用户消息列表）等查询方式。Twitter API在查询接口上做到了高度的统一：

- **规范HTTP接口**：使用GET方法作查询操作，使用标准的HTTP Status Code作错误返回。
- **请求参数统一**：无论是哪个查询接口，制定查询用户、输出条目数等的参数都是统一的。
- **输出结构统一**：查询接口的输出即是资源表达，没有其他多余的信息——例如绝不会出现错误号之类的信息。同时所有资源表达都可以输出等价的JSON和XML表达，方便不同开发平台的开发者使用。

Twitter本身是一个很简单的应用，但其API接口已多达近100个。学习使用这些接口是一个很大的挑战。但正是这样的设计方法：划分资源、定义统一资源表达、使用规范协议、设计一致查询接口让这变成了极为简单的事。统一的接口还让交互代码能够轻易重用，开发客户端库的工作量也变到最小——实际上完全不需要专用的客户端库。在这样的规范模式下，接口也更易保持稳定，再加入新功能，也能保持统一的风格。

这种设计方法可被归为RESTful的设计方法，实际上Twitter的初始版本使用宣称RESTful的Ruby on Rails开发，熟悉RoR的同学能很容易地在API的URL模式上看出它的痕迹。关于RESTful，我们不在这里展开叙述。Twitter并未完全运用RESTful的所有设计思想，但仅划分资源、设计统一接口就已带来了大量的好处。

相对于优美的查询接口，Twitter的写接口就稍显凌乱。RESTful设计思想中，希望能整理出对资源的统一操作，如增删改，这样对资源的写操作也能如查询一样简单统一。但在实际项目中，把纷繁复杂的业务操作（例如retweet等）归进一套统一的操作语义中相当困难，Twitter很努力地希望实现这一点，甚至在Lists接口中引入了最“规范”的RESTful接口形式，但仍无法完全做到。

Twitter的API并未遵循其他通用规范，这与遵循AtomPub的Google Data Protocols不同。但有趣的是，Twitter的API渐渐成为各种微博产品API接口的实际标准，具有了相当的通用性。比如新浪微博的API接口，与Twitter几乎完全相同，将一个Twitter的客户端改造成新浪微博客户端，是件非常容易的事。

前端Web界面

有了强大稳定的API服务，前端开发将更加自由和简单。在新的浏览器下，我们可以对如何实现前端界面做新的思考。

刚开始时，浏览器的能力很弱，充其量是一个文档展示器。此时Web应用的主要任务就是提供文档，供浏览器展现。此类Web应用用上了最简单的所谓MVC模式：从Model拿出数据，然后交给View——一个HTML模板，渲染成HTML，由浏览器展现就可以了。这是开发者的“The good old days”，程序结构简单清晰，维护简单。

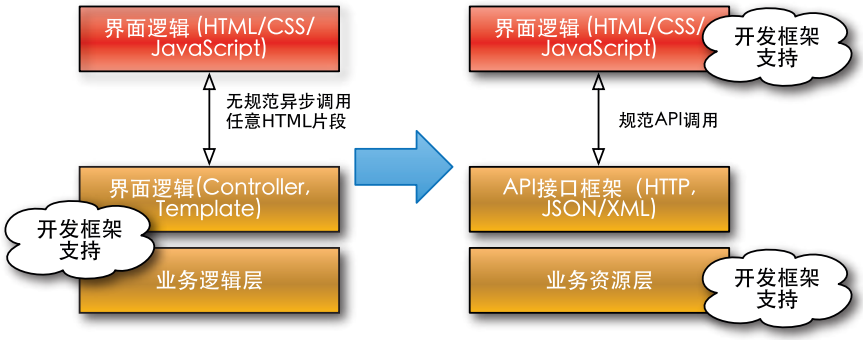


图2　界面开发框架支持示意图

然而，这样的体验显然无法满足用户要求。人们开始使用更多的客户端脚本和Ajax技术，实现更丰富的体验，Web应用的架构也要适应这样的变化。这就引入了一个令人相当纠结的问题——什么逻辑应放在客户端，什么逻辑应放在服务端？这个问题会有很多答案，也有像GWT一样的方案来自动解决这个问题。当然在实际项目中见到的，更多是临时AD-hoc方案，哪里需要异步交互体验，就在哪里开个接口，服务端和客户端的程序结构都渐趋混乱。

一体化的用户界面，需要在客户端和服务端用两种语言配合实现，还有联调查错，其痛苦程度可想而知，我们自然会有完全在客户端实现用户界面的想法。这个想法已有大量尝试，但成功者不多。我想，主要原因是：

- 浏览器能力的限制，浏览器的功能和性能都无法支持实现完整客户端。
- 服务端接口困难，业务逻辑接口无统一模式，交互困难，无统一有效的安全机制，难以开放出公共网络，由浏览器端直接调用。

正如我们所看到的，一个设计良好的开放API服务解决了第二个问题，而新的浏览器大战正加速解决第一个问题。开放API服务已提供了业务所需的所有接口，不必再为前端界面增加开发——也就是说，我们完全不必为用户界面开发增加服务端开发量；而新浏览器的功能和性能足以支持非常复杂的功能；HTML5的Offline能力，也使我们不再担心大规模客户端代码等造成的下载时间和流量问题。

我们需要关注的，仍是应用的复杂性。以往如jQuery等框架把主要精力放在简化页面DOM操作、快速实现交互效果这一方面，对应用复杂性的关注并不多。Twitter开发团队在文章中就提到了这方面的挑战，就如在ROR等框架中一样，他们在客户端设计了一个页面路由系统来管理页面状态。各JavaScript框架对如MVC模式的支持、事件的统一管理和分发、界面状态管理、有效利用HTML5特性等方面的探索还处在初始阶段，我们应能在不久的将来看到更多的成果。

总结

我们面临着全新的环境：更多的使用Web应用的方法，全新的浏览器能力。过去的许多假设都要被打破，对互联网架构和技术，我们要有全新的思考。

面对这样的变化，许多应用的架构都会趋向“API服务+富前端”的模式。实现这个模式的首要挑战在实现API服务。因缺乏有效的模式和强有力的约束，特别是缺乏开发框架的支持，当前的Web开发项目容易存在分层不清、接口难以提取等问题。直接的产品需求逼迫我们解决这些问题，使用RESTful的设计思路，能帮助我们有效地理清业务资源，围绕业务资源设计简单、一致的接口，在良好的接口之下，也能更有效地整理和组织业务代码。

浏览器能力的增强和拥有完善接口的API服务的建立，使我们可以趋向于完全使用客户端技术实现界面逻辑的模式。而随之而来的问题是，客户端应用规模迅速增大，复杂性上升，我们需要一个能有效控制复杂性、提高开发效率的客户端框架。

阻止你的MySQL集群罢工

——MySQL高可用性方案探讨

■ 文 / 杨海朝

作者简介：

杨海朝，新浪首席DBA，在大规模高并发、海量访问特别是大规模数据库运维方面有丰富的管理和维护经验。热衷于数据库设计、性能优化、分布式部署方案和高可用性方面的研究。曾就职于康盛创想，从事大访问量网站的部署以及优化工作，2007年末加入新浪负责整个公司的数据库管理工作。

MySQL在企业级应用中越来越多，随着企业把一些重要的数据迁移到MySQL上，对MySQL的高可用性就提出了很高的要求。特别是在互联网应用，7x24小时的业务需求随处可见，这就要求MySQL的部署是高可用的，首先是不存在SPOF（Single Point of Fail），其次是在出现故障的情况下，能保证业务持续稳定的运行。

MySQL高可用方案概述

通过MySQL的Replication结构比较容易实现Slave的高可用，例如：通过部署多台Slave，在这些Slave前面加上负载均衡的软硬件（F5、LVS、Haproxy等），配合监控Slave Replication状态的一些脚本，能实现Slave的高可用，在某一个Slave出现故障时，自动从提供服务的Slave池中去掉这个有故障的Slave，业务层感觉不到这个故障的存在，但Master的高可用不像Slave那么容易实现，基于业务对一致性的要求目前大多数公司对MySQL服务的写入方式都采用集中写，在企业里更多的是对Master可用性的需求。

MySQL目前存在的高可用解决方案

目前MySQL存在图1中的几种高可用性方案。

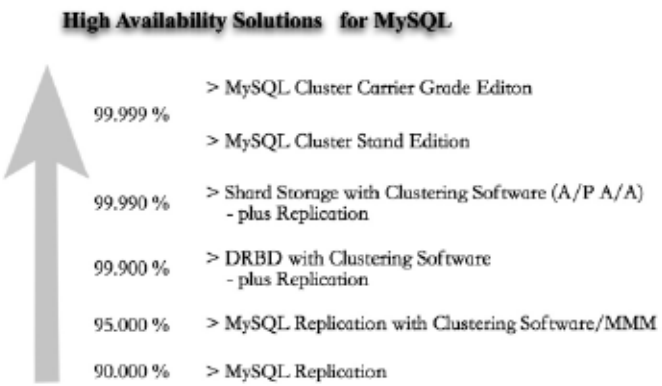


图1 MySQL高可用性方案和SLA的关系

不同的方案能提供的可用性级别也不同，可以根据业务的重要性以及出现故障影响的范围选择合适的可用性方案。MySQL Replication能提供一个9的SLA（service-level agreement服务品质协议）。MySQL replication与一些集群软件结合使用能实现一个9、一个5的可用性级别，例如：MMM（Multi-Master Replication Manager for MySQL）或Wackamole。Heartbeat+DRBD+MySQL通过底层的块设备复制能实现3个9的可用性级别。MySQL的共享存储方案包括A/P（Active/Passive）和A/A（Active/Active）两种方式能实现4个9的可用性级别。MySQL Cluster的标准版（SE）和电信级版（CGE）能实现5个9的SLA。下面针对上面这几种高可用解决方案进行详细说明。

MySQL高可用性方案之MySQL Replication实现

MySQL Replication复制结构如图2所示。

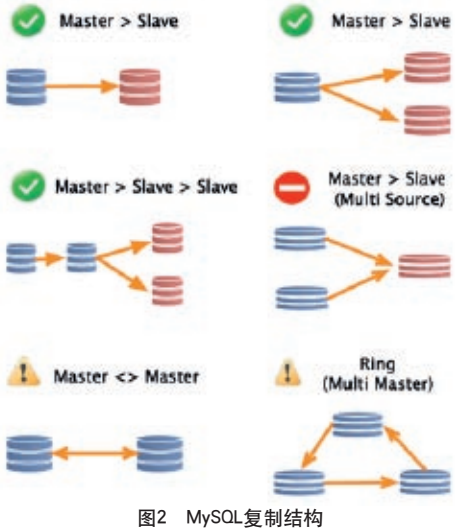


图2 MySQL复制结构

目前MySQL复制支持：一台Master带一台Slave；一台Master带多台Slave；一台Master

带一台或多台Slave（或称为中继Slave），这台Slave再带一台或多台Slave（树状结构）；也支持两台MySQL Server互相复制（也称作双向复制）和环型复制，但这两种结构在使用过程中需要注意多点写入的自增和更新丢失问题；MySQL当前不支持多台Master带一台Slave结构，但可以通过双向复制结合Blackhole引擎变相地实现两台Master带一台Slave的结构（如图3所示），或者利用Federated引擎来实现多台Master带一台Slave的结构（Multi-Master and a Single Slave）（如图4所示）。

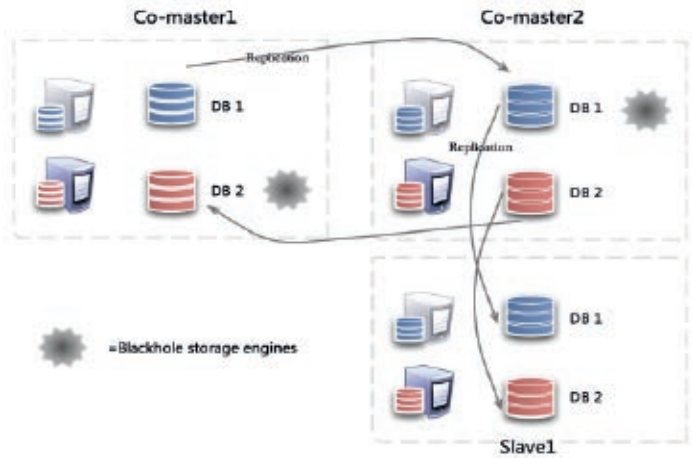


图3　双向复制结合Blackhole引擎变相地实现两台Master一台Slave的复制结构

图4　利用Federated引擎变相地实现多台Master一台Slave的复制结构

就像前面已经提到的通过在Slave前面加一些类似F5、LVS、Haproxy等负载均衡软硬件能比较容易地实现高可用，在Master出现故障时，提升某一台Slave为Master来替换掉有故障的Master继续对外提供服务，同时需要所有Slave（除了提升为Master的Slave）重新从新的Master开始同步数据，这个过程很容易出现数据丢失，同时怎么保证选择的Slave的数据是最接近故障Master的也是一个需要考虑的问题。目前大多数公司都是采用这个方案，把提升Slave为Master以及修改其他Slave 重新从新的Master同步数据的操作步骤写成脚本，在收到报警时，人工的执行这个脚本，实现故障切换。这个方案只能实现一个9到两个9的SLA，如果你的业务承受不了一年内有3~4天的宕机时间，那么可以考虑在成本和可用性做出权衡的能实现99.900%的Heartbeat+DRBD+MySQL方案。

MySQL高可用方案之Heartbeat+DRBD+MySQL

Heartbeat+DRBD+MySQL方案是通过DRBD（Distributed Replicated Block Device，分布式复制块设备）与Heartbeat结合实现99.900%的SLA。MySQL不断更新的数据文件通过DRBD实现底层的块数据同步，在提供服务的机器（Active Server）出现故障时，备节点（Passive Server）的Heartbeat检测到主节点没有心跳（例如：Ping不通主节点），备节点自动接管虚拟IP，同时挂载DRBD分区，启动MySQL服务，其他的Slave节点通过虚拟IP（作为所有Slave的Master_host）继续复制，不需要做成本较高的修改所有Slave从新的Master同步的操作，同时在大多数情况下能避免在一台Master带多台Slave结构中因切换造成的数据丢失问题，详细结构如图5所示。

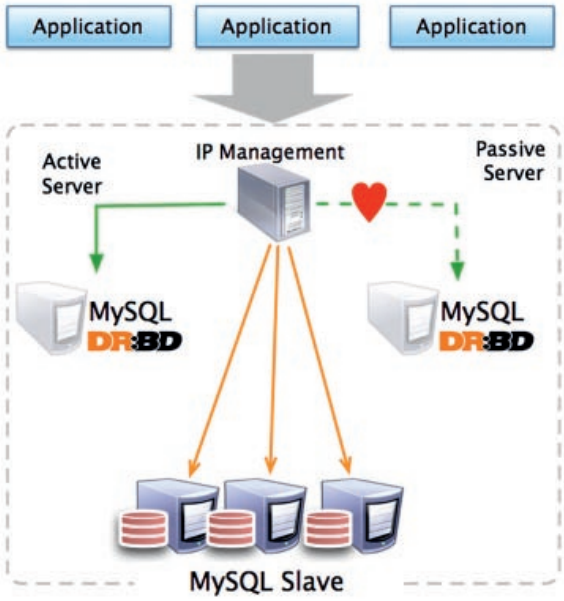


图5　Heartbeat+DRBD+MySQL replication结构

使用这套方案需要有一定的硬件资源投入。备节点是不能和主节点同时挂载DRBD分区，同时要求备节点性能和主节点相当，能承受主节点能承受的压力波峰，同时这台备节点在出现故障之前处于闲置状态。这个方案不能达到亚秒级（Sub-Second）的切

换时间，切换时间与心跳检测参数设置，文件系统检测时间和表的恢复时间有关。表的引擎最好采用事务型引擎，减少Failover需要的时间。此方案存在众所周知的裂脑问题，可以通过增加心跳线路，适当的心跳重试参数以及结合Fence（Stonith）设备来减少出现的概率。很多人担心使用了DRBD导致MySQL性能下降的问题，我的观点是所有的方案都有一定的应用场景，一个方案不能解决所有问题。DRBD的性能已经表现得相当不错，即将加入Linux内核的主干代码。如果你遇到性能问题，首先考虑的是Sharding策略，如果你对性能和可用性要求都很高，那这样的业务一定是非常核心产品，建议投入一定的硬件成本使用下面所说的MySQL共享存储方案。

MySQL高可用方案之共享存储

MySQL共享存储方案要求有相对比较高端的存储设备，这个方案能实现99.990% 的SLA。原理是把可变数据都放到一台存储设备上，多台Server共享这个存储设备，MySQL服务和前端应用之间的通信使用G比特的公共网络，MySQL服务和存储设备使用FC私有网络，两个网络独立互不影响。FC交换机和存储设备都是两台，在一台设备出现问题时自动使用另一台设备，一台存储设备和另外的一台存储设备自动Mirror（保证存储的可用性问题）。一台MySQL服务器出现故障，切换到另一台MySQL服务器上，继续提供服务。因为不需要像Heartbeat+DRBD方案中做一系列检查，挂载分区，恢复表文件和启动MySQL服务等操作，所以切换速度非常快，能达到亚秒级的切换时间。例如部署两台MySQL服务器（Active/Passive）、FC交换机和SAN设备（自动做mirror）各两台，结构如图6所示。

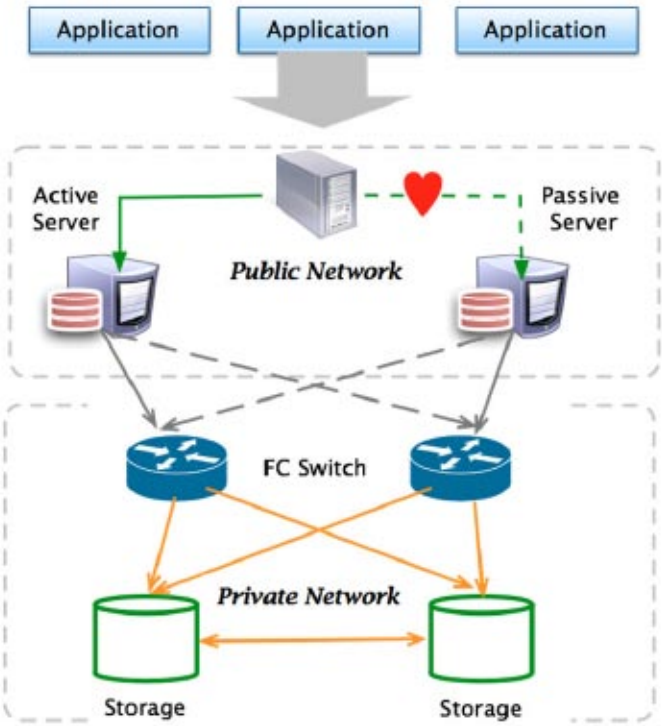


图6　MySQL共享存储方案结构

如果需要两台或多台MySQL Server在同一时刻都能对数据进行读写，需要类似GFS一样的分布式文件系统配合。这个方案非常类似于Oracle的RAC结构，数据库可用性提高的同时硬件成本投入也指数级增加。在MySQL的实际应用中，目前国内的互联网公司使用这个方案的为数不多，使用的都在非常核心产品上，通过购买EMC、NetApp、3PAR等存储设备来实现。这个方案除了需要投入昂贵的硬件成本外，扩展性有限，同时在多IDC容灾部署上也不能很好地满足不断增长的业务需求。

MySQL高可用性方案之MySQL Cluster

MySQL Cluster是之前的MySQL AB（现在属于Oracle公司）在2003年收购了爱立信（瑞典）的一个部门，把这个部门开发的用于电话系统的NDB引擎加入了MySQL形成的。它是一种Shared-Nothing结构，由数据节点、MySQL节点、管理节点三个组件组成，这种结构也是未来发展的趋势。如图7所示。

MySQL Cluster能实现亚秒级（Sub-Second）的Failover，能够达到99.999%可用性，一年的宕机时间控制在5分钟以内。MySQL Cluster支持在线添加节点来扩展服务的容量，这些操作完全对应用透明，同时也具有类似于Proxy的功能，能实现真正的Connection Pool，开发人员不需要关心读写访问地址的区分问题，也不存在SPOF。MySQL Cluster使用的是NDB（Network Database）存储引擎，它容忍多个数据节点和MySQL节点同时出现故障，允许数据节点出现故障的个数依赖于NoOfReplicas参数的设置（每份数据的副本个数），同时能在出现故障时通过重新配置集群来屏蔽掉这些故障节点，它的Geographic Replication特性能解决跨IDC的容灾和高可用性问题。在五个MySQL节点，四个数据节点和两个管理节点的MySQL Cluster系统允许的最大故障节点如图8所示。

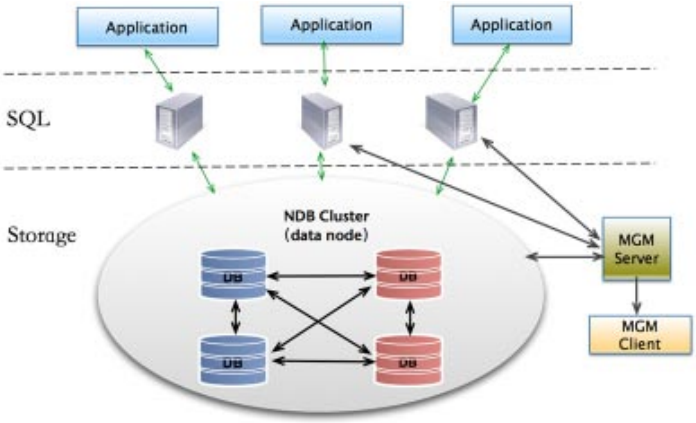


图7　MySQL Cluster结构

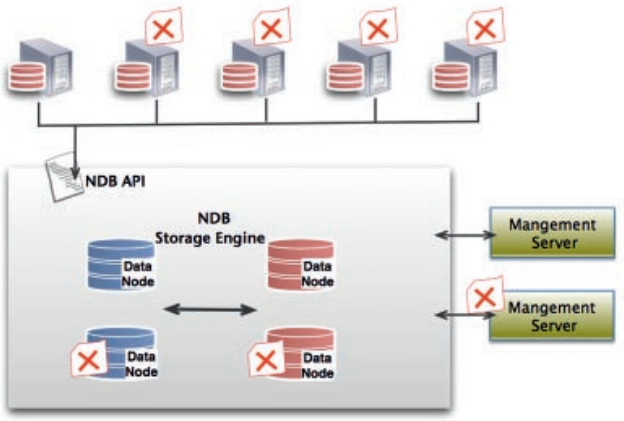


图8　MySQL Cluster最大允许的故障节点

几种MySQL高可用方案的比较

每种高可用方案都有相应能达到的SLA级别，通过上面几种可用性的方案介绍，这里归纳出这几种高可用方案在可用性方面的几个特点：

可用性需求项目	MySQL Replication	MySQL Replication +Heartbeat/MMM	MySQL Replication +Heartbeat+DRBD	MySQL共享存储	MySQL Cluster +Geographic Replication
IP自动Failover	否	是	是	是	否
DB自动Failover	否	否	是	是	是
Failover消耗时间	依赖用户执行的切换脚本时间	依赖Cluster软件Fs和Table的恢复时间	小于30秒	小于10秒	小于3秒
Slave自动数据同步	否	否	是	是	是
地理冗余支持	是	是	需要结合复制	需要结合复制	需要结合复制

MySQL Slave的高可用实现较容易一些，主要是保证Master的高可用。之前的MySQL AB（现在是Oracle公司）把Heartbeat+DRBD+MySQL作为他们向企业提供的重要收费服务项目之一，由此也说明了此方案的具有很实际的用途。MySQL共享存储部署简单，但硬件投入大的特点被许多企业不能接受。MySQL Cluster虽然具有非常高的可用性，但因为存在业务场景方面的限制，所以使用的公司也不是很多（具体可以查看相关文档）。目前Heartbeat + DRBD配合MySQL Replication来实现高可用的MySQL集群是企业的首选。对于单个MySQL实例来说，高并发量和大数据量两个因素避免同时存在，如果同时存在再加上对高可用性要求，那么需要投入的维护和硬件成本就会很高。如果是非常核心的业务，可以考虑MySQL共享存储方案，毕竟对可用性要求很高的核心业务对成本就不能考虑得太多。这里需要说的是任何一个方案都有其使用的场景，不能解决所有问题。在选择的时候结合自己的应用特点进行选择，我只是给出一些大致的指导思想。选择高可用性方案也是在业务需求和成本之间的一种权衡，同时如果能结合应用一起考虑可用性问题，能实现更高级别的高可用。

MySQL高可用方案有很多路要走

怎样能使一个5x8的DBA能支持多个7x24的在线业务？如果你是公司的DBA，正在参加团队建设游玩得高兴的时候或上下班的路上，你是否愿意接到电话或者报警告知某台数据库宕机，需要立刻进行处理？你是否想拥有以下场景：你的MySQL集群在遭遇任何故障时，都能完全自动Failover，不需要人为干预；在收到某台MySQL服务器的报警或接到电话时，大多数情况不用做任何操作，只是为了知道有故障存在，在非常规故障发生时能通过发送一条短信实现MySQL服务的自动切换。目前很多公司的MySQL集群系统都做不到这里描述的场景，MySQL集群系统的高可用还有很多路要走。

深入浅出数据仓库平台统一架构

■ 文 / 蒋杰

作者简介：

蒋杰，支付宝BI首席架构师，主要研究方向为分布式数据库、行业数据建模、高并发查询系统架构设计、实时数据仓库。个人博客 http://www.binotes.net。

随着公司业务迅猛发展，BI系统支撑能力无法及时满足业务发展需求，数据仓库系统建设大量功能模块，但模块利用率低下，无法对BI系统建设进行准确有效的效果评估等问题，困扰着我们系统架构者和规划者。

随之而来面临的各方面压力如：

- **业务部门**：需求提了好多天，怎么还没做出来？
- **需求分析师**：需求太多了，你的需求得排到半个月之后了。
- **公司领导**：每年都投了那么多钱，数据仓库建设取得的实际效果如何？
- **开发人员**：天天加班，啥时候才是头啊？

BI系统价值的反思

BI系统作为企业基本信息系统之一，它的价值是显而易见的。但在大多数情况下，建设部门和业务部门的关注点不同。BI建设部门（如数据平台部）只关注数据仓库系统各种能力的建设、改进和提高，这种能力恰恰是业务价值的基础。业务部门则从业务价值角度考虑，以对BI实际工作促进为目标，结果导致BI系统评价的偏差。

这个偏差也是很多数据仓库项目失败的根本原因，其实业务需求是支撑BI系统建设最有效的催化剂，而这种BI需求以前仅仅只是内部客户（业务部门），而现在也有一直来自于外部客户（如商户或者大客户）的数据查询、数据分析需求，如何了解好业务的需求、弄清楚来自客户的最大痛处，如何为客户提高效率、控制客户实施风险，这是我们系统建设者和规则制定者在系统建设前期必需下的工夫。

经过反思以后应该去关注的问题有：

- 如何选用一个完整的解决方案来满足快速增长的数据？
- 如何选用高效的数据库来满足数据仓库的应用？
- 如何让基本价值和业务价值得到统一？
- 如何规划BI系统的长期行进路线？
- 如何建立廉价、高扩展性的系统架构？
- 如何建立自有开发标准？

BI平台行业解决方案

经过几年BI系统建设，我们慢慢摸索出了自己的一套BI行业解决方案，来支持大型数据仓库建设，经过归纳，BI行业解决方案可以分为下面八个部分。

- **数据分发中心方案**：Gaea系统。
- **底层的解决方案**：混合数据仓库模式、自定义SQL标准。
- **ETL调度系统方案**：Schedule Server系统。
- **数据模型设计方案**：DW五层数据模型、CDW架构。
- **统一门户方案**：数据门户平台。
- **元数据管理方案**：元数据平台。
- **监控和审计方案**：监控和任务平台。
- **用户帮助系统方案**：知识库平台。

综上所述，只有八者合一，才能构成完整的企业级行业解决方案。

数据分发中心

在大型企业环境下，因为不同业务应用系统建设的时间不同，所采用的数据库也不同，所以对于数据仓库的数据来源也不同，这给我们进行数据抽取和装载带来一定的困难。

下面来看一下支付宝数据来源情况。

- 不同类型源数据获取：源数据库有Oracle、SQL Server、MySQL、PostgreSQL等，源文件有带分隔符文件、定长文件等外部文件导入。
- 每个数据库或者数据文件字符编码不同，

为了在目标数据库做数据准备必须进行转码操作。

- 需要在规定的时间窗口内，完成大规模增量数据抽取。
- 在线数据实时接收。

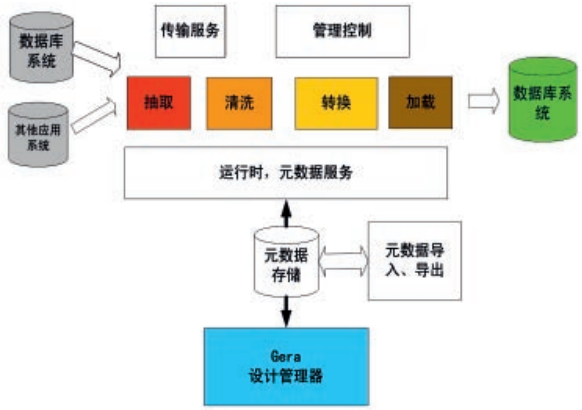


图1　数据仓库建设所需要解决和克服的问题

经过多年实践的积累，为了适应快速企业应用的变化、快速部署新增应用和进行系统扩展，我们建设了一个数据分发中心系统，包括以下基本功能。

- 图形化灵活开放的配置模式。
- 数据抽取数据快，完全基于源数据API实现。
- 流水化作业方式——抽取和转载。
- 功能权限控制。
- 配置元数据管理接口。
- 分布式数据导入、导出。
- 接口扩展性好。

混合数据仓库模式

混合模式数据仓库架构的设计原则：发挥各自优势和特性来满足业务需求。具体分为下面三个层次。

- **分布式计算层**：对结构化和非结构化数据进行大批量并行计算。
- **业务逻辑层**：实现高可用性，复杂业务的计算和数据压缩存放。
- **前置应用层**：提供前端查询接口和高并发的应用。

架构如图2所示。

为了方便开发和后续维护，支付宝使用了自定义SQL标准——阿里SQL（简称：Alisql），其目的是用一套代码实现跨平台和异构数据库。

Alisql业务设计特点包括以下方面。

- 对开发人员来说不用关心底层是什么数据库，透明的数据仓库架构来实现异构数据库之间差异的转换。
- 节约开发成本，对开发人员来说不用需要特别去了解底层具体是什么数据库，同时也不需要了解具体数据库之间差异问题。
- 统一的代码管理机制，有利于元数据管理。

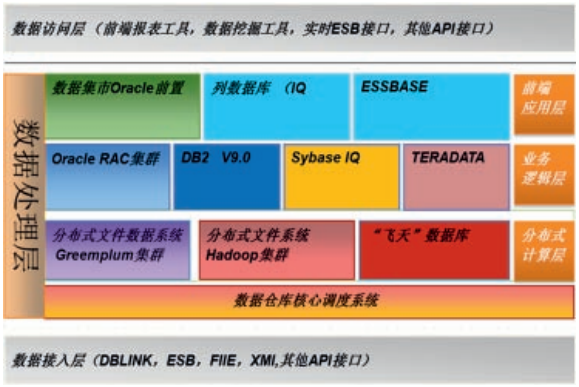


图2　混合模式数据仓库架构图

Alisql技术设计特点包括以下方面。

- 为了支持业务系统数据的可扩展性和灵活性，程序访问数据的方法，必须使用Alisql的函数调用，以保持编写的代码与数据库无关，这样做可以为数据仓库提供完整的数据平台，为将来的可扩展数据提供了灵活的接口。
- 程序出现的SQL应该符合SQL92标准，对于DB2、Oracle、Greenplum、Sybase和Teradata中不符合SQL标准的SQL函数或用法，应使用Alisql提供的函数调用，如果已有函数不能满足要求，应该扩展Alisql编写新的函数以封装各种数据库语言的差异。
- 根据支付宝特殊的业务需求，可以封装自有的特定函数给数据仓库工程师或者分析师使用，从而提供了更多灵活的性能。

ETL调度系统

传统作业调度方式的问题，一般来说在UNIX/Linux系统上会以Cron Job方式进行定时作业调度，而在Windows系统上则以内建的排程工具进行作业调度。

以操作系统提供的调度工具来进行作业的执行时常会面临到以下一些问题。

- 作业运行时间设定不够弹性，修改不易。
- 需重新执行时必须登入系统进行处理，作业本身须处理日志保存。
- 日志分散在各服务器上，查看不易且权限控管容易有漏洞，不容易做到多台服务器间的多项作业执行顺序控制。
- 复杂作业流程无法以系统调度工具达成，须再以程序来控制流程的执行，作业本身须处理通知功能，设定修改不易。

数据模型设计

数据仓库系统建设中由于不同企业内部业务不同，他们的数据模型设计也不同，而重点区别是在物理模型上。

但我这里要强调的是，不要去盲目直接设计物理模型，这会造成业务归纳不全面，导致后续物理模型设计复杂，如何梳理好企业概念数据模型（CDM）更为重要。概念数据模型的内容包括重要的实体及实体之间的关系，不包括实体的属性，也不用定义实体的主键。概念数据模型的目标是统一业务概念，作为业务人员和技术人员之间沟通的桥梁，确定不同实体之间的最高层次的关系。

大家在具体项目实施的时候，可以参考业务IBM业务概念间最初的关系，所有业务信息都是可以用九大概念的词汇来表示，从而使概念框架提供了一套通用的结构，它描述了所有业务环境，并且每一种信息概念都可用三个分层来详细说明：分类分层（是什么）、描述分层（有什么）、关系分层（做什么）。

通过概念模型梳理，支付宝数据仓库数据模型采用五层架构设计，从而来满足不同业务需求，如图3所示。数据仓库层次结构图细分如图4所示。

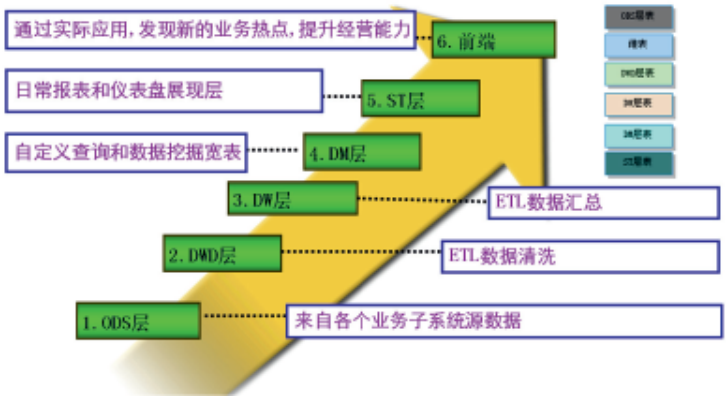


图3　支付宝数据仓库数据模型的五层架构设计

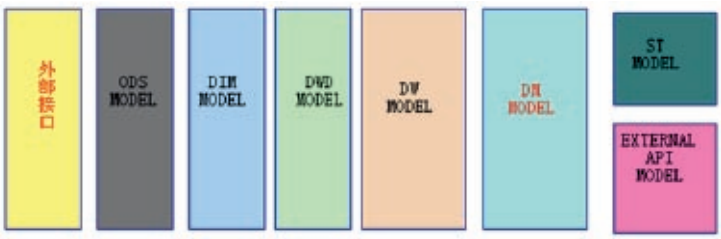


图4　数据仓库层次结构图细分

下面说明一下层次结构。

- **外部接口**：指从业务系统抽取数据接口。
- **ODS Model**：即操作数据存储（Operational Data Store，ODS）层，它是从业务系统过渡到数据仓库核心层的操作数据的模型，ODS模型的数据结构与业务系统基本保持一致，并永久保存。
- **DIM MODEL**：即维表层（DIMENSION MODEL），它是指专门从ODS层抽取及维护各个维表，在更新完维表之后再进行下面各个层次的清洗。
- **Data Warehouse Detail Model**：是数据仓库核心层数据模型之一，用于存放完整详细的历史数据，目前基本按照第三范式设计。其设计目标是为后续的Data Warehouse Model提供灵活性和扩展性的基础，并把部分字段转换成维表代码，以减少存储，同时可以在DW层无法支持需求时直接为应用层提供数据。DWD层不涉及跨主题/跨业务系统整合，实体关系与业务系统基本相同。由于与业务系统耦合程度较高，其稳定性会受到业务系统的影响。
- **Data Warehouse Model**：存放详细历史数据的数据仓库层，目前按照混合模式设计。数据仓库层是核心层，设计目标是为StarSchema（DM/Report/External）三个应用层提供足够的灵活性和扩展性的基础。同时，由于它包含了整个数据仓库的大部分数据，是数据仓库项目的基础，所以保证该层数据长期的稳定性是首要的。
- **DataMining Model**：在DW基础上，用于组织DataMining的宽表数据层或者特殊需求查询数据层，它们的首要目标是最大程度地满足业务的灵活需求，实现所谓的“商业智能”。
- **ST Model**：在DW基础上，用于生成报表的数据层。
- **External Model**：在DW基础上，提供外部直接访问数据仓库的外部数据层，把数据仓库内部结构屏蔽掉，只提供外部需要访问的数据。

统一数据门户

统一数据门户就是有效地将企业中的各种数据、商务过程和应用结合起来。我们内部采用BO作为报表展现工具，为了更好地与门户融入，基于BO的API进行二次开发集成来无缝集成到统一数据门户。

采用Java开源系统成熟框架来进行获取、分类和组织，建立索引，并提供转换、过滤、聚合和检索的能力，实现信息的目录和内容的有效管理，从而让用户更方便地获取这些信息。

同时，根据实际情况，我们内部需要集成数据分析平台、内部需求管理平台、权限系统服务、数据报告服务、OLAP多维分析等内部应用平台和工具。如图5所示。

元数据管理

虽然在数据仓库环境中元数据极为重要，但是要无缝地将所有部分的元数据集成起来却是一个相当困难的任务。这是因为：

- 建设元数据不是为了元数据而去搞元数据，要

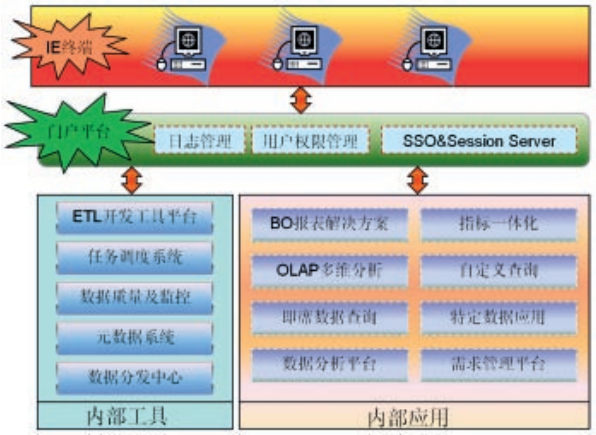


图5　统一数据门户方案示意图

让元数据不仅作为数据身份证，还要延伸开来成为数据的轨迹，进而形成一张大网把展现、抽取、建模等数据操作过程网起来；

- 元数据管理涉及到多角色的协调工作、企业的数据标准以及企业的数据架构；
- 前端应用越简单越好；
- 在数据从源系统到准备区，再到数据仓库的过程中，元数据没有一个公认并简易的传递方法；
- 在数据仓库中保持统一的元数据版本控制的工作相当繁重，而且很困难。

因此，元数据在数据仓库中拥有非常重要的地位，元数据的内容就好像是数据仓库的DNA，修改它们的内容直接影响到数据仓库的行为。同样调整数据仓库的行为也是通过调整元数据的内容来实现的。

元数据应用系统架构分为5层，如图6所示。

- 源系统层：元数据主要提供者，涵盖DW、业务类系统。
- 采集桥接器：结构化、非结构化元数据的采集分析接口。
- 元数据存储层：元数据存储时元数据管理实现的核心，存储层为功能应用提供信息支撑。
- 功能管理层：元数据的维护接口、实体关联度分析接口、版本管理接口、变更通知接口、实体查询接口、影响分析接口、血统分析接口、导入导出接口、实体差异分析接口、元数据访问配置接口、管理接口。
- 访问接口层：元数据的维护、实体关联度分析、版本管理、变更通知、实体查询、影响分析、血统分析、导入导出、实体差异分析、元数据访问等展现。

监控和审计

监控主要分为两大块：系统监控和数据质量监控。

- 系统监控主要有硬件故障监控、数据库故障监控、调度系统任务运行状况。通过的动态柱图、清晰展示故障统计。

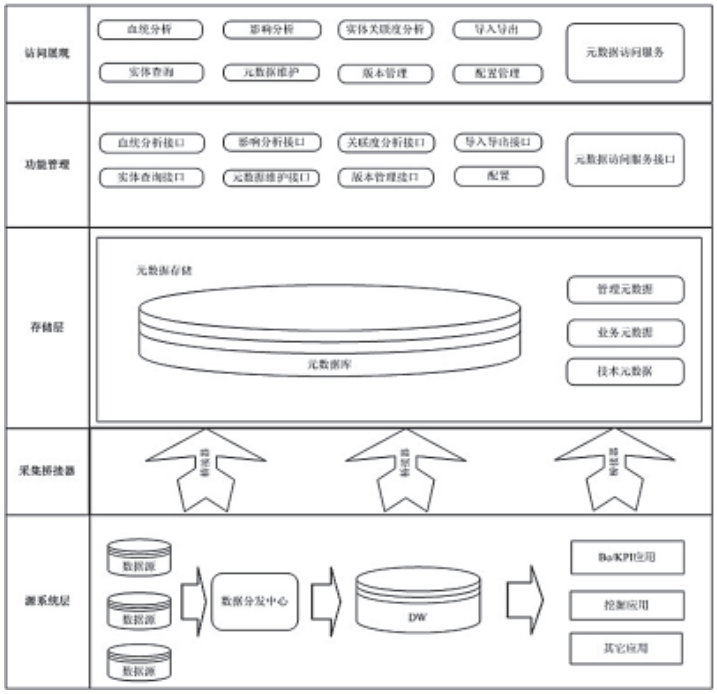


图6　元数据应用系统架构示意图

- 数据质量监控主要是维表NULL值检查、Email字段非法检查、IP非法检查、业务逻辑恒等式检查、记录缺失检查、记录重复性检查、数据一致性检查、记录数比对检查、时间逻辑检查等功能，目的是为了提高数据准备性。

用户帮助系统

用户帮助系统是为了实现内部客户在线信息检索，我们通过知识库系统，提供丰富的站内搜索及帮助，如图7所示。

图7　支付宝的用户帮助系统

总结

随着Web2.0技术日益成熟，云计算技术广泛应用，数据仓库平台技术也已经发生很多变化，不是仅仅局限传统关系型数据库，小型机，高端存储的体系架构，而是分布式数据库，廉价PC架构所取代。更多开源BI项目出现，并且在企业得到应用，使得业务成本大幅下降。

消息队列漫谈

■ 文 / 穆荣均

作者简介：
穆荣均，美团网研发总监。2005年毕业于清华大学，先后供职于百度、饭否。

消息队列，最初作为进程间通信的一种机制出现，在同一个操作系统内工作。

当不同的主机通过网络连接起来组成一个分布式的系统时，在不同的主机之间传递消息的需求就是显而易见的了。这通常是通过消息队列系统来实现的。

企业应用中的消息队列

软件巨人IBM、微软等公司对这个问题的答案是，各自推出私有的面向消息的中间件产品，集成在各自的中间件产品或操作系统中。

IBM在1993年就推出了对这一问题的解决方案MQSeries，2002年更名为WebSphere MQ。微软在Windows95、Windows98、Windows NT4.0中集成了Microsoft Message Queue组件，并持续发展至今。

在2001~2002年间，J2EE平台的一个API标准Java消息服务（Java Message Service，简称JMS）出台，确定了J2EE平台的应用组件间创建、收发消息的规范。JMS在企业领域获得了广泛的应用，IBM、Oracle、之前的Sun以及JBoss等厂商的消息队列服务均支持JMS。

在2004~2006年间，摩根大通和iMatrix公司一同推动了一项旨在实现一个厂商中立、开放的消息队列协议的工作，其成果便是AMQP（Advanced Message Queue Protocol，高级消息队列协议）。和JMS相比，AMQP是一个进步。JMS是一个API级别的规范，没有规定消息的格式，因而不同厂商的服务器和客户端产品可能不能互通；而AMQP同HTTP、SMTP等应用层协议类似，确立了服务器和客户端的通信协议，按照此规范实现的服务器和客户端确保可以相互交互。

iMatrix完成了AMQP的第一个实现——OpenAMQ，并协助摩根大通完成每日几亿消息量的应用迁移工作。另外的开源实现包括基于Erlang的RabbitMQ，以及基于C++的Apache Qpid。这两套系统的幕后东家分别是VMWare和Red Hat。

JMS、AMQP其实都主要集中在企业应用领域，处处都是IT巨头们的身影。不同的厂商希望用协议来规范互联互通的标准，这些消息队列服务器的使用者们也不希望陷入某家公司的私有标准中。

互联网世界的消息队列

与强调规范、兼容，同时也常常意味着复杂、臃肿、性能偏低的企业领域相比，互联网世界则是百花齐放，各家可能选择完全不同的实现，同时也通常更加轻量、性能更出色。

Twitter的消息队列

Twitter架构是一个典型的使用消息队列的架构。依靠消息队列，用户发表的Tweet得以分发到其Followers的Timeline中。

2008年初，Twitter开源了其消息队列服务器Starling。Starling基于Ruby语言，通信协议令人耳目一新的直接采用了memcached协议的Set和Get语义，这大大的简化了消息的生产者以及消费者客户端的编写，可以直接使用丰富的memcached客户端库，也自然而然地解决了分布式问题。

Starling 的持久性（Durablity）是靠Binlog机制保证的，每次Set或者Get，都把操作及上下文信息记录到Binlog文件中，Binglog文件大小超过某个阈值时将被清空（清空时确保消息队列是空的），服务重新启动时自动重放一遍Binlog中的操作即可恢复内存中的队列。

然而，虽然Starling本身不会丢失消息，但是在整个系统中，消息仍然可能丢失。比如，消息的消费者从消息队列Get了一条消息，那么Starling服务器上这条消息将不再存在；如果客户端还没有处理完消息就崩溃了，这条消息就从整个系统中彻底地消失了。

Twitter自然也意识到了这个问题，其在2008年底开源的Kestrel就是Starling的进化版。Kestrel使用Scala语言重新实现以增强性能，更重要的是其增加了消费消息时的ACK机制。不过memcached协议并没有这种语义的命令，Kestrel是通过扩展Get命令的文本字段来实现的。

但是这种ACK机制可能导致同一条消息被消费多次：如果客户端已经处理但是在确认某一条消息之前崩溃，该条消息会被重新加入队列并被重新处理。这是一个必要的设计权衡，系统设计应处理消息重复的情况。

此外，在分布式部署环境里，多个 Kestrel 服务之间并不保证消息的严格有序。这实际上是大多数分布式服务的特点。

国内技术界的开源贡献

或许是受Starling采用memcached协议的启发，在Twitter开源Starling之后不久，新浪的Steve Chu在其分布式K-V存储系统MemcacheDB的基础上开发了MemcacheQ。

MemcacheQ底层使用BDB的Queue存储模式，也因而引入了一个限制：所有的消息按照固定长度存储。这要求在系统设计之初预估消息的最长长度，超过此长度的消息将无法插入。如果消息长度短于该固定长度，则会用空白符补齐。所以，如果某系统消息长度变化很大，比如一个文章长度差别很大的日志系统，是不适合采用MemcacheQ作为其消息队列系统的。

对于限制消息长度为140字的微博系统来讲，这一局限影响就不大了。据新浪微博的架构师Tim Yang介绍，新浪微博采用了MemcacheQ作为其消息队列服务器，并部署了多个MemcacheQ实例，生产者客户端根据memcached协议自然地把消息分布到多个服务器，消费者客户端轮询多个服务器获取消息以避免单点故障问题。

金山的张宴于2009年底开源了一个基于HTTP协议的消息队列服务器HTTPSQS，使用了Tokyo Cabinet来做数据的持久化存储，HTTP服务使用Libevent提供的evhttp框架解决。从作者公开的资料看，HTTPSQS已应用于金山内部多个系统中。

不过系统设计者在考虑是否采用MemcacheQ和HTTPSQS时，应当注意这两个消息队列服务器自身都没有提供机制解决客户端崩溃时消息可能丢失的问题。

异步架构及其他

消息队列的背后实质上是一种“异步处理”的架构思想，可以说今天的所有分布式应用都是异步的。在有的复杂系统中，已经不能简单地使用一个单一的消息队列服务器来解决问题，典型的例子是Facebook。

在Facebook，提供消息传输的系统叫做数据高速公路（Data Freeway），主要由网络日志服务器Scribed、Hadoop分布式文件系统以及一些配套工具集等组成。Scribed日志采集终端部署在Facebook的上万台应用服务器上，中间用于汇聚、归类、传输的服务器也有数百台。

数据高速公路系统完成了用户动作事件的汇聚和传输，最终把消息输入Feed系统中，其他例如数据分析系统、实时搜索系统等也依赖于数据高速公路的数据输入。在Facebook如今超过5亿用户、消息数据来源于上万台服务器的规模下，数据高速公路每秒钟传输的数据超过1GB，消息延迟不超过15秒。

美团网的消息队列实战

在一个脚本语言+MySQL搭建的规模不大的系统中，其实有更加简易的消息队列实现，用一个数据库表加上一个简单的类封装即可实现。在美团网初期，我们使用这种直接操作DB来模拟的简易消息队列解决了邮件、短信等服务的异步处理。比如，设计数据库表如下：

```
CREATE TABLE 'eventqueue' (
'id'int(10)unsigned NOT NULL auto_increment,
'ctime'int(10)unsigned NOT NULL default'0',
'status'tinyint(3)unsigned NOT NULL default'0',
'topic'varchar(32)NOT NULL default'',
'data'blob NOT NULL,
PRIMARY KEY('id')
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

其中字段data是消息序列化之后的结果，方便起见，序列化可以是JSON。topic标志了本条消息的类型，也便于客户端识别本消息是否需要关心。

使用status字段标志消息是否已被消费而不是直接删除，因为我们很有可能在遇到Bug时需要恢复

现场或者重新处理某一段已经消费过的消息。而为了避免DB无限制的膨胀，可以设计简单的Rotate机制，比如一个星期一个数据库表，仅保留最近几个星期的数据。

插入消息很简单，在DB中直接insert一条记录即可。消费消息时，客户端取下一条状态是“未处理”的消息，处理成功后标记为“已处理”。这种ACK机制是可以保证消息在系统中是不会丢失的。

不过上述设计存在诸多限制，比如仅支持一个客户端工作。如果我们基于此做相应的改进，比如采用多个客户端按照topic取消息等，是可以在一定程度上解决问题的。但本质上这不是一个分布式的方案，不具备松耦合、高可用、可扩展等特性。不过在实践中，这种简易方案通常是快速解决问题的一把利器，我们使用类似上述的简易消息队列解决过多个系统初期的消息传输。

美团网目前每天发送几十万邮件、几万短信（2010年9月数据），使用简易DB消息队列虽然仍可以解决问题，不过其给DB引入不必要的负载，不方便分布式部署，很难实现高可用等问题开始凸显。我们需要一个更好的消息队列系统。在诸多的开源解决方案中，我们最终选择了Beanstalkd。

Beanstalk协议也是类似memcached的文本协议。Beanstalk协议原生的支持了ACK操作：先保留（Reserve）一个任务（Beanstalk把一个消息被称作为一个Job），成功处理后删除（Delete）该任务，处理失败也可以释放（Release）该任务；如果客户端超过一个设定的TTL后没有响应，则服务器自动释放该任务。Beanstalk简化的任务生命周期如图1所示。

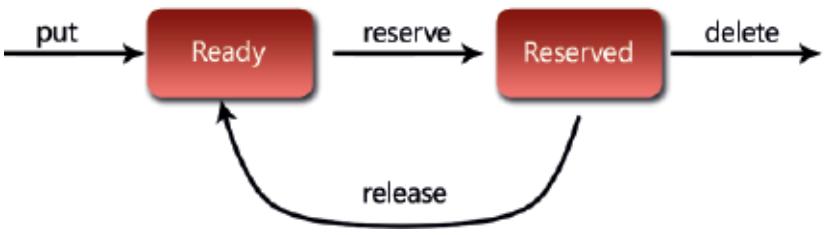


图1　Beanstalk简化的任务生命周期

Beanstalk内置支持多个队列，客户端可以同时监听（Watch）多个队列，任何一个队列有新的任务都会响应客户端的保留（Reserve）请求。

此外，任务是可以设置优先级的，因而实际上Beanstalk提供的是优先队列。这非常切合我们的实际需求，因为短信网关发送速度有限，对于用户主动触发的短信我们期望以最高的优先级发送；而对于系统触发的短信比如订阅短信，我们可以以较低的优先级发送。

至于分布式，和memcached一样，Beanstalkd是依靠客户端来实现的，也不保证消息的严格有序，这一点和 Kestrel 完全一致。

Beanstalkd应该说也是经过考验的产品了，开发者社区也非常活跃，大部分客户端都是其他开发者提供的，你甚至很容易找到监控队列状态的各种脚本。美团网基于Beanstalkd构建的邮件、短信队列也稳定运行，在我们现在这个规模下，内存占用不到40MB，CPU几乎没有负载。

回顾及展望

可以看到，在企业应用领域，强调互联互通，因而大家致力于统一的协议规范；而在互联网领域，则以满足实际需求为主要标的，实现方式不拘一格。不同的业务、不同的数据量、原有系统的主要构建环境，都可能影响工程师们的选择。

我们来看看一些其他的互联网应用是如何使用消息队列的，以供大家参考。基于Java架构的Linkedin，自然地选择了ActiveMQ作为消息队列。eBay在解决搜索索引的实时更新时，采用了一种基于多播（Multicast）的消息传输机制。在一次TUP沙龙上，前139技术总监王鹏云提到了一种基于Topic订阅，并可以实现事件回放的消息队列，应用于139微博以及百度的一些系统中。据传淘宝有自己实现一个消息队列系统Notify并有计划开源。此外，iMatrix现在鼎力支持的ZeroMQ也值得我们关注：它并不是一个软件，而是一套接口类似于Socket的开发库，使用这套库可以很容易地在系统中集成各种类型的消息传输功能。（注：本文成文于2010年9月底，您看到这篇文章时，其中引述的数据可能有更新的。）

门户网站负载均衡技术的六大新挑战

■ 文 / 李晓栋

记得上大学时，我和好友老郭讨论最多的话题便是：“像新浪这样的网站是如何支撑如此巨大的访问量？”也曾通过各种手段，猜测新浪服务器的数量、操作系统和应用软件的版本……一切都是那么神秘。毕业那年，有幸加入新浪，终于一点点地揭开了这层神秘的面纱。2004年某厂商设备介绍会上，我初次接触到了负载均衡技术。之后的几年时间，可以说是负载均衡设备在网站推广的黄金爆发期。

发展到今天，一方面硬件设备依然保持了强劲的实力，另一方面以LVS、Haproxy为代表的软件负载均衡也异军突起，被人们所认可。在新浪，软、硬件负载均衡并存的格局已有三年多的历史了，除了既往积累的经验外，近一年来，我们也看到了负载均衡所面临的一些新挑战，在此跟大家分享。

挑战一：Web应用对七层交换的依赖度越来越大，显著增加了负载均衡器的压力。

七层交换技术的引入，极大地解放了架构师和程序开发人员，同时也使我们越来越习惯依赖于它，甚至直呼上瘾。很难想象，如果没有负载均衡器的话，现有Web架构中的大量需求应该如何实现？在充分享受其便利性的同时，我们也看到了一些隐忧。一方面越来越多的流量正从四层交换转为七层交换；另一方面七层交换的规则也越来越趋于复杂。在双重作用下，负载均衡器的压力急剧上升。对于任何一台负载均衡器来说：支撑相同的请求量，七层交换所消耗的CPU要远远高于四层交换。特别是在瞬间高并发连接的突发流量面前，负载均衡器面临着严峻的挑战。

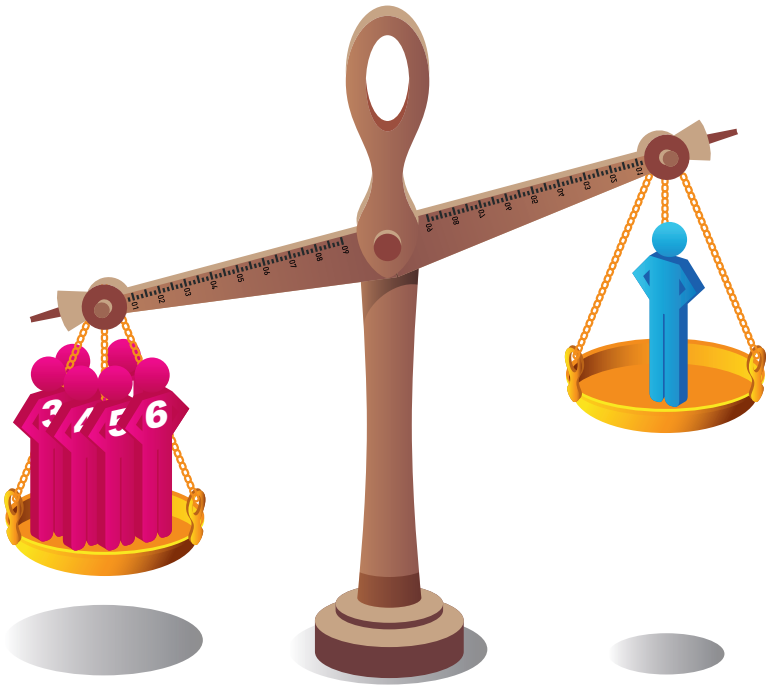
挑战二：微博等互联网新兴产品的出现，对负载均衡器的运维工作提出了更高的要求。

微博不仅改变着亿万网民的生活，而且也正悄然推动着运维体系的建设。

首先，与传统的新闻、博客相比，微博用户对服务质量的敏感度更高，而且这种敏感度会伴随着一次次的“@”和“转发”传播扩散。在过去，当用户访问新浪服务感到慢时，反映的渠道多是打客户电话。而现在只需要在微博上一个简单的“@” 就可以与新浪的客服和技术人员直接沟通。作为流量进出的一个关卡，当微博等线上关键业务出现访问异常或故障时，工程师们都迫切地想知道：是负载均衡器的问题吗？此时故障诊断的效率显得至关重要。在实际工作中我们发现：单纯依靠负载均衡器提供的CPU、内存、连接数等统计信息，还不足以发现一些隐蔽问题。传统的抓包分析耗时耗力且效果不佳，再加上有些故障现象与客户端、后台服务器上的某些特殊设置有着千丝万缕的联系，所有这些交织在一起，给我们故障诊断带来了不小的挑战。例如有次我们发现：负载均衡器偶尔会给客户端返回HTTP 5xx的响应，当时特想快速地知道究竟是什么样的HTTP请求会触发这样的现象。但可

作者简介：

李晓栋，新浪网研发中心基础架构部技术经理、集团金牌讲师。在新浪有6年多的系统、网络、安全相关工作经验，现负责新浪网络设备和操作系统相关研发工作。自2007年9月以来带领团队研发了新浪DDOS防火墙、广域网加速器、软件负载均衡管理系统、网页挂马检测系统等重要产品，为公司节约成本上千万元。



惜的是，负载均衡器上仅有统计数字而没有请求的完整记录。在花了很大力气抓包分析后，最终定位到是由于后台一个PHP程序不小心给页面设置了一个错误的HTTP Header，导致Web Server的HTTP响应不能被负载均衡器所接受，最终给客户端返回5xx。因此在故障诊断方面，我们需要有更先进的理念和手段。

其次，微博在国内正处于快速成长期，会随时根据访问量来灵活调整服务器的数量和系统架构。在这种快速灵活的变化面前，负载均衡器相关的配置调整工作也随之增加：频繁的上下线服务器、变更七层规则等。面对这种情况，我们需要思考：如何能更加快速安全地完成好这些变更、如何能避免工程师每天被动地陷入这些重复烦琐的工作中等。目前一些硬件设备提供了API接口，像增删Server、调整Server 权重等这类风险性极低的操作可通过API接口操作，以达到提高效率的目的。而Haproxy、LVS则缺乏这样的 API接口，需要单独开发。

除此之外，关键应用对负载均衡器的监控也有越来越多的新需求。比如：有些应用希望当负载均衡器检测到服务器池中活跃的服务器数量少于一定比例后，便提前给系统管理员作出预警；及时发现服务器池中权重等设置不合理的问题等。

挑战三：多核处理器时代下，Haproxy等用户态的软件负载均衡正面临新的性能瓶颈。

近几年来CPU发展进入了多核时代，CPU由过去的单核发展到四核、六核、八核、十二核，甚至更多，而主频则变化不大。在这种趋势下，充分利用多核特性显得尤为重要。但在我们研究中发现，像Haproxy这类基于用户态的软件负载均衡，其对CPU主频的依赖度要远远高于CPU核数。换言之，在高主频、核数少CPU下的性能很有可能要优于低主频、核数多的CPU。这一点，在Haproxy服务器选型时尤为重要。据我们分析，这主要是由于操作系统对多核（多CPU）下的并发支持度还不够好。

挑战四：软件负载均衡发展路上的“鸡蛋-篮子”理论的艰难选择。

硬件负载均衡器往往以单台高性能著称，而Haproxy、LVS为代表的软件负载均衡的优势则在于成本低廉、可灵活定制，其性能与服务器CPU、网卡等硬件直接相关（当然特殊的优化也很重要）。正如前面提到的，当七层交换流量越来越大时，我们究竟是该投入成本让单台LVS、Haproxy足以支撑如此大的流量，还是让更多中等性能的服务器共同分担这些流量呢？这就是所谓的经典的“鸡蛋－篮子”理论：究竟该不该将鸡蛋放到一个篮子里呢？

其实不同的选择各有利弊。2~3年前，我比较赞同将流量分摊到多台软件负载均衡器上，当时主要考虑到风险的分散。而现在，我更倾向于将流量集中于一台上。之所以这样，是从以下四个角度考虑的。

第一，目前国内IDC内每个机架所放服务器数量跟电力配额直接相关。而负载均衡由于其特殊性，往往是两台为一组，这样每增加一组都会增加额外的电力开销，特别是在电力资源紧张的IDC内，提高单台软件负载均衡器的承载能力可以为关键业务腾出更多的机架来。

第二，从服务稳定角度考虑，我们通常会将LVS、Haproxy直连核心交换机，如此一来，每增加

一组，就意味着会占用更多的核心交换机端口资源。

第三，是基于管理成本的考虑。LVS、Haproxy之所以能在新浪得到广泛应用，较低的管理成本是重要原因之一。在新浪我们通过一套集中管理平台和快速初始化的办法实现了运维成本的非线性增加。但不可否认的是，每新增一组，运维成本或多或少总会增加一些。之前也曾设计过一套“同机房内负载均衡器的集群池方案”，即：在一个机房内主备机的数量比不再固定为1:1， 虚拟IP（VIP）会根据集群池中每台Haproxy/LVS的负载状况，动态地“漂”在其中某台上。但后来发现这个“听起来很美”的方案，在实际运行中遇到了种种问题，运维成本不降反升。最终我们又回归了传统的1台Active+1台Standby的模式，正所谓简单即是美。

第四，目前硬件负载均衡器正朝着“更高的性价比”方向发展，换句话来讲，如果我们不提升软件负载均衡器的单机支撑能力，则终有一天，其与硬件设备相比的成本优势将会淡去。

挑战五：在新时期下，如何找到负载均衡的最佳软硬结合之道？

朋友、同行聚会时，常有人问我：“你们有了Haproxy、LVS后，会不会不买硬件设备了？”、“你最近又在山寨什么？”每次听到这些，我都会微微一笑。如前文所述，Haproxy、LVS这类的软件负载均衡和硬件设备各有优势，在我看来，负载均衡的“软”、“硬”解决方案并非水火不容，只要找到最佳的软硬结合之道，鱼和熊掌还是可以兼得的。下面是我们在长期摸索中，总结出来的一些经验。

软件负载均衡可优先承担四层交换流量，让硬件设备更专注于七层交换：由于工作方式和原理的不同，专注于四层交换的LVS在稳定性、单机支撑能力、易维护性、管理成本等多方面均要大大优于Haproxy。特别是在DR模式（即单臂）下， 单台LVS足以应对绝大多数业务的访问量。

优先保障“明星”产品占用宝贵的硬件设备资源：这里指的“明星产品”是指那些用户群正处于快速增长，并被广泛追捧的热点互联网产品，例如微博。考虑到负载均衡器一旦发生异常或宕机后，将对产品的美誉度和用户体验产生一定程度的影响，这属于无形成本的损失。正所谓“好钢要用在刀刃上”，我们可以优先将这类流量放到硬件负载均衡器上。

对于必须采用七层交换的重点服务来说，尽量避免同一重点服务的流量全部放在软件负载均衡器上。例如某重点服务分布于四个IDC内，则可考虑两个IDC内使用Haproxy，另外两个IDC使用硬件设备。这样一方面可以在一定程度上规避使用Haproxy可能带来的风险，另一方面也方便对软、硬件负载均衡器的稳定性、响应时间等进行长期对比观察。

要充分利用好同一IDC内的软、硬件负载均衡器，当一方负载高时，另一方可协助其分担流量，缓解燃眉之急。

总而言之，在负载均衡方面的支出正所谓该花则花、该省则省，合理使用可以让你在保障服务稳定的前提下，获得最佳的投入产出比。

挑战六：软件负载均衡器的资源复用，在降低成本的同时，同时也面临着一定的运维风险。

目前我们的软件负载均衡器分布于全国各地，其中一些中小规模IDC内的软件负载均衡器的负载并不是特别高，而这些机房普遍又需要VPN、自动安装等服务，单独为这些服务再放1~2组服务器显得很不划算。因此我们想到对软件负载均衡器进行资源的复用，即：在软件负载均衡器上同时运行VPN等服务。在实际中发现，这种资源复用面临两方面的风险：一是VPN、自动安装、负载均衡可能分属于不同的管理员，这样大家对同台服务器进行操作会增大因配置冲突、操作不当等导致的服务间互相影响的概率；二是非负载均衡的服务可能会突发占用过多的CPU或网络资源，对正常的负载均衡服务造成了一定的影响。由于LVS、Haproxy服务的特殊性，像Xen这类通过虚拟化来实现资源隔离的办法又不太适用；对服务器流量进行QOS设置，虽然可以起到一定的效果，但配置方面还是有些烦琐。还有更好的办法吗？这确实值得我们思考。

当然除了以上六方面挑战外，负载均衡领域还有很多值得研究之处。不同网站有不同的实际情况，希望本文能对大家起到抛砖引玉的作用。

一种与众不同的游戏分发平台的技术架构

■ 文 / 张福

作者简介：

张福，麻球游戏技术总监，原永恒之塔技术负责人，在技术团队建设，技术管理，系统架构方面有非常深厚的经验，同时在游戏安全领域有所建树。

麻球游戏的使命是为开发者建立一个无处不在的能通过游戏赚钱的游戏分发平台，与一般的平台相比，它具有独特的商业模式。这种商业模式使得麻球游戏在产品设计和技术架构上都具有独到之处。

独特的模式

麻球游戏采用的是一种聚合内容、再分发内容的模式。这个模式有四种角色：Flash游戏开发者、广告主、渠道和平台本身。

这四种角色是结合在一起的：Flash游戏开发者接入麻球游戏API，然后上传游戏到游戏；麻球游戏通过渠道将游戏分发到游戏玩家面前；广告主在平台上进行广告投放，广告被推送到各渠道的各个游戏中，然后展示给玩家。

尽管我们的游戏分发网络提供了大量游戏，但由于平台本身不承载内容，所以绝大部分用户甚至都不知道我们平台的存在。玩家在渠道上玩游戏，会看到广告主投放的广告，小游戏开发者也会通过广告播放来获取收入分成：越多人玩你的游戏，广告播放的次数就越多，你的广告收入分成就越多。所以这就导致我们平台的技术也非常有特点：虽然有众多的内容、渠道、开发者和活跃用户，但它本身所使用的服务器和带宽都非常低，这是因为平台是聚合内容后再把这些内容分发到各个网站上，这些网站承担了服务器、带宽、访问所带来的开销，而我们只需要承担往网站分发内容的开销即可，所以麻球游戏的这套技术体系的服务代价非常低。

数据分析

既然麻球游戏只需要承担分发内容的开销，那是否就可以说我们处理的数据量并非很大？事实上，我们处理的数据量非常大，数据处理主要集中在广告方面。因为所有收入都来自广告，并且给渠道和开发者的分成也是通过广告收入进行分成，所以广告决定了整套商业模式能否顺利运转。

为了提高广告的收益，就需要准确定位广告人群。因此麻球游戏有一个强大的数据后台进行数据分析，可以非常精确地给不同属性、不同地区的用户推送合适的广告。广告投放的流程如下：

首先，广告主通过自助式后台进行广告投放，投放时可以指定费用、CPM或CPC、目标人群的偏好、环境数据等，然后广告主还可以制定一些限制和策略，比如专门投一个广告让所有在特定网站上玩我们游戏的人看到这个广告，还有就是广告所投费用如何花费（每天定额花费还是尽快消耗广告费）。定义好这些之后，系统会产生一个广告订单，这个订单会被传送到广告服务器，然后广告服务器把广告发送给合适的人。

怎样知道合适的人呢？我们是这样做的：用户在某个网站上玩我们的游戏时，那他所在的网站、时段以及游戏类型等信息将被传送到广告服务器，当广告服务器收到这些信息后，就开始逐条分析，而分析的过程是在后台服务器上完成的：服务器会根据以往的历史数据、每个人的访问记录及各种策略以及我们对这次广告请求所能搜集到的数据，进行运算对比，最终在广告库中选择一个合适的广告，返回到广告服务器上，然后再将其推送到Flash游戏中，这样玩的人就可以看到适合的广告。为了能够尽量给用户推送合适的广告，我们有一整套分析用户行为的系统，这个系统由数据仓库、数据分析服务器以及一套数据抽取收集程序构成，每天这套系统都会分析几十GB的用户行为数据，然后生成新的规则提供给广告服务器使用。

基于Erlang语言框架的个性化搭建

麻球游戏所使用的技术、软件和系统在国内应用极少，还不太为国内技术人员所熟悉，但相信会在未来几年热门起来。

我们的核心模块采用了Erlang语言来进行开发，作为一门开发语言，它还包括很多框架性的内容。使用Erlang来做我们这套技术架构有诸多好处：开发速度快，只需要做逻辑上的开发，框架的部分包括怎样处理网络连接、怎样分发任务、怎样实现容错，都不必关心。此外，Erlang还有几大特点：第一，Erlang是现有语言中对并发处理支持最好的，因为它最早应用于爱立信的核心交换机上，所以其并发处理能力远高于常规技术；第二，Erlang能够实现动态的代码升级和热替换，在以前系统升级时我们要停掉线上服务，把升级程序替换掉之后再启动，服务是有一个中断的，而Erlang能实现代码的热替换，不停机就可以实时更新代码，这也是迫于当时核心交换机的要求，因为核心交换机永远不能停机或者停止服务；第三，Erlang具备强大的容错能力和高稳定性、高并发性和高扩展性。正是基于Erlang语言框架的这些特性，我们实现了一个非常强大并且稳定的系统。

此外，由于我们的广告系统采用的是Flash内置广告，因此充分利用了Flash的一些特性。通过和Erlang后台配合，Flash游戏在运行过程中会不断地与后台进行数据通信，我们就可以追踪到玩家玩游戏的动作和各种情况，从而进行有针对性的广告推送。

运维管理

我们非常关注用户玩游戏的体验，比如要花多长时间打开游戏和广告等。这些性能数据会被实时传送到后台，并被归类到数据仓库进行分析，所以我们能够知道不同地方、不同设备、不同时段，用户访问我们内容的情况，包括有多少人访问、感觉怎样、延迟有多少、有多少失败的，我们都会对这些性能数据进行整理和监控。我们所有的服务器都有传往中心的监控系统，监控的内容不仅仅是服务器是否宕机这么简单，我们会监控很多细致的内容，比如从各地区来的流量怎样、各地区访问游戏的类型和数量怎样、各地区用户体验和延迟怎样。监控的内容会被细分到用户体验的各个方面，然后我们再进行处理。

这其中，涉及相当多的技术架构，我重点介绍一下自动化管理方面。我们使用自动集中管理工具Puppet来管理整套服务器，它可以在中心服务器上对所有的服务器进行配置和应用管理，是应用管理、分发管理和配置管理的一种非常好的实践。Puppet的中心服务器上保持了各种应用及其信息，其他服务器的一系列状态都会保存到中心服务器上。假设某天几百台服务器都坏掉，我们可以实现在短时间内将整套服务重新部署到全新的机器上。我们这套技术架构和管理方式结合在一起，实现了优越的配置管理。自动化管理的架构如图1所示。

我们全球的系统都是使用同一个Puppet master来进行中心化管理，它上面有非常多的配置文件，分别对应着不同的客户端。为了管理大量复杂的配置，我们对配置文件进行了抽象，并一级级继承下来，举个例子：我们要配置2台DELL 2960服务器，分别作为nginx服务器和NFS服务器。首先新建一个配置文件DELL 2960，这个配置文件描述了DELL 2960服务器所需要默认安装的所有软件和需要进行的配置，然

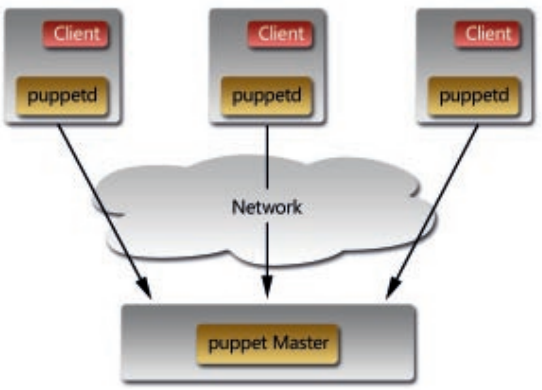


图1　自动化管理架构图

后分别创建DELL 2960-nginx和DELL 2960-nfs，这两个配置文件继承DELL 2960，然后再加入各自特有的配置。通过这样的抽象然后分层管理，可以降低复杂度，同时管理成千上万的机器。

除了Puppet外，我们还使用了PXE和IPMI来帮助我们安装和管理大量的服务器。我个人认为，PXE+IPMI+Puppet堪称目前自动化部署和应用管理的最佳实践。

结束语

未来互联网架构的发展趋势已经非常明显：第一，向云的方向发展，也就是往强大的资源管理，避免浪费的方向进行；第二，向基础架构和业务架构分离的方向发展，随着业务的市场和需求的灵活变化，需在保证基础架构足够稳定和强大的情况下，灵活支持上层的业务需求。整体的发展趋势就是这个样子。相信麻球游戏会在这个发展大潮中走出自己的一片天地。

基于动态内容的缓存加速技术

——F5 Web Accelerator 产品技术剖析

■ 文 / 徐超

作者简介：

徐超，F5北方区售前技术顾问，主要负责与客户就系统架构设计、应用交付技术进行交流和沟通的工作。曾任职于TOM.COM运维部门，有丰富的运维经验。带领或参与了多个重要项目：无线增值业务MySQL数据库系统部署及调优、TOMCDN部署及运维、推出VIACDN商业平台、Eachnet平台的部分架构设计和MySQL数据库的调优。

本文将对F5 Networks的Web Accelerator产品进行技术原理分析，介绍其如何对动态内容进行缓存和处理、如何实现缓存内容的及时更新、如何提升页面的打开速度。

由于权限所在，只能从售前层面了解到产品的功能以及根据实验测试等方式，总结出相关技术及实现方式，希望可以起到抛砖引玉的效果。

Web Accelerator的技术亮点

Web Accelerator是F5推出的对动态内容（如使用PHP、Java/JSP、C#/ASPX等程序动态输出的内容）构架下的Web应用进行加速的产品，除了拥有常见的应用负载均衡功能外，对于缓存技术的贡献，主要有以下两个亮点：

IBR（Intelligent Browser Reference，智能浏览器引用技术）——最主要的功能，实现对动态内容的缓存和及时更新，本文将着重介绍；

MultiConnect（多连接技术）——辅助功能，实现对浏览器连接数限制的突破，本不属于缓存技术内容，但是可以在Cache系统中实现，就一并做个简单介绍。

智能浏览器引用技术

访问路径上的缓存

从我们打开浏览器到一个页面的完全打开，这期间有多少个缓存在为我们的访问进行加速？

- 浏览器本地缓存；
- 局域网出口缓存/代理；
- ICP反向代理/缓存。

传统的缓存（Cache）技术主要集中在后面两项，如大名鼎鼎的Squid就既可以做局域网出口的缓存也可以做ICP反向代理。Web Accelerator则专注于上面的第一项和第三项。

Web Accelerator认为承载动态内容的页面每次产生变化的只是一部分内容，而大多数内容不会发生变化。如果可以只传输变化过的内容，而不变的内容直接从浏览器缓存中获取，则既可以提高动态页面打开速度，还可以节省网络流量。

RFC 2616与缓存相关的标签

在RFC 2616中定义了与缓存控制相关的标签，凡是使用到这些标签的应用自然都会利用和遵守这些标签的规范。以下几个标签在缓存控制中非常重要，我们前面提到的三处缓存都会严格遵守这些标签的定义：

- **Last-Modified** 标记目标的最后更新时间；
- **Cache-Control**：
  - private 私有内容不允许被缓存（缺省）；
  - public 公共内容可以被缓存；
  - must-revalidate 每次使用缓存拷贝必须向服务器验证；
  - no-cache 不允许被缓存；
  - max-age 可以被缓存的时间，以秒为单位。
- **Expires** 如果目标被缓存，在此时间之后失效；
- **Content-Length** 指示目标的长度；
- **Transfer-Encoding: chunked** 指示目标将以chunked方式传输；
- **Pragma: no-cache**同Cache-Control: no-cache；
- **HTTP 200 OK** 指示一个正常的HTTP响应，其后附带完整的目标内容；
- **HTTP 304 Not Modified** 指示目标并未发生改变，可以使用发起请求方的缓存拷贝，其后不附带任何内容；
- **If-Modified-Since** 出现在HTTP请求中，如果目标的最后修改时间大于/晚于这里指定的时间，则服务器返回HTTP 200并附加完整的目

标内容，否则服务器返回HTTP 304。

在RFC标准缺省设定下，什么内容可以被缓存？

- 必须有正确的Content-Length值；
- Cache-Control中必须包含public，而又不包含private，max-age大于0；
- Expires指示的时间必须在将来；
- Pragma: no-cache 和Cache-Control: no-cache不可以出现在HTTP响应中；
- URL中不得包含“?”。

我们发现HTTP200与HTTP304的响应有巨大的区别，前者是完整下载目标内容，后者只有一个HTTP响应头，HTTP304必须由一个带有If-Modified-Since标记的HTTP请求来引导。

浏览器什么时候才会发起带有If-Modified-Since标记的请求呢？

凡是浏览器自身的缓存里保存有目标的有效拷贝，浏览器就有可能会自动发起IMS请求，如：

- 按下浏览器的“刷新”按钮；
- 在地址栏输入曾经打开过的网址；
- 本地拷贝的内容必须含有Last-Modified标记。

注意：<Ctrl>+<F5>不会产生IMS请求，而且会附带Cache-Control: no-cache标签。

为什么动态页面无法使用缓存

传统的缓存技术为每一个目标在本地的拷贝设定了一个失效时间，只有一个拷贝失效了，缓存才会向原服务器询问是否有新的版本。由此导致原服务器的内容可能更新了很多次，可是通过缓存访问的结果是只看到了最初的版本和最后一次更新后的版本。

通常一个页面由一个HTML输出作为载体，引导其他部件（CSS/JS/XML/JPG/GIF）的加载。动态内容的HTML输出通常没有Content-Length和Last-Modified，带有“Transfer-Encoding: chunked”和“Cache-Control: private, no-cache”，Expires设定为一个过去的时间等各种不允许被缓存的标记；而其他部件通常是一个文件，并不是由程序运行之后输出的，在一段时间内它们维持不变，可视为静态内容，但是他们会由于后台操作或者用户的操作产生变化，但是文件名却不变！因此只有等待浏览器缓存内的拷贝过期后，才会获取到新的内容。

因此为了能让浏览器在任何内容产生更新后都能获得最新的内容，我们必须让所有的内容输出的时候都带有“Cache-Control: no-cache”。显然，这样做的后果是浏览器的缓存完全失效！同样的，对于Squid等传统的缓存系统也存在这样的问题。

通过上面的解释我们知道，由于内容更新的随机性和实时性导致我们不允许浏览器与服务器之间存在任何的缓存，因为这些缓存会让更新不能及时表达。因此动态内容的开发者，会通过设置HTTP标记的方式让所有的缓存失效。

如果我们想在访问路径上部署一个缓存，并让它生效，我们将面临三个问题：

- 如何修改HTTP 标记让浏览器缓存生效；
- 如何判断一个页面的内容更新了，并找到哪些内容更新了；
- 如何让缓存器内的拷贝总是与服务器上的一致、如何让浏览器每次访问都可以拿到最新的内容。

让浏览器缓存能够为动态页面工作

Web Accelerator会根据我们配置的策略对动态内容的以下HTTP标签进行控制：

- Transfer-Encoding: chunked 和 Content-Length；
  - 通过一次性接受所有chunks，重新拼合形成unchunk内容
  - 计算unchunk长度自动添加Content-Length
- Last-Modified；
  - 使用最后一次保存此目标的时间
- Expires；
  - 删除并添加Cache-Control: max-age
- Cache-Control: max-age；
  - 设定为我们指定的值
- Cache-Control: public。
  - 删除private增加public

这样处理之后，浏览器就可以开始缓存我们的页面及相关内容了。在Web Accelerator的策略编辑器中，我们可以根据URL的部分或者全部进行匹配，绑定不同的HTTP标签控制策略。

Web Accelerator如何识别动态页面的更新并保持更新

对于首次访问，Web Accelerator是完全地缓存所有从服务器下载的内容，并且把这个HTML内包含的各个部分的内容都下载到Web Accelerator里，并按照我们定义的策略对HTTP标签进行处理。

从图1可以看到，Web Accelerator在接收到HTML内容后，会对HTML内容进行扫描，并且预先下载HTML内包含的其他元素，如：CSS/JS/JPEG/GIF等内容。

当Web Accelerator把HTML内包含的元素都下

图1　首次访问的处理步骤

载完成后，会为HTML增加Last-Modified标签和改写Cache-Control等标签，以形成浏览器能够缓存的内容，然后再把HTML发送给浏览器，此时浏览器就能够把动态HTML内容进行缓存了。浏览器会继续加载HTML内包含的各种元素，由于Web Accelerator之前已经把这些内容下载到自己的缓存中，所以这个过程不会再去原服务器进行下载了。

以后每当Web Accelerator收到一个对HTML内容的请求，首先要从原服务器下载一份，与本地保存的进行对比，同时还要把这个HTML包含的各个部分都向服务器发送IMS请求，以验证每一个部分是否存在更新。由于我们在第一次请求中，返回来的HTML信息中含有must-revalidated，所以每次浏览器要使用缓存中的内容都需要跟服务器端进行验证，就是发送一个IMS请求。WebAccelerator收到一个请求后，首先要检查这个请求的内容是否是一个HTML内容，如果是一个HTML内容，还需要检查HTML内容所包含的各个元素，都要从原服务器进行下载后验证，或者向原服务器发送IMS请求来验证。

Web Accelerator会对服务器返回的内容进行判断，以确认页面是否更新，页面更新有三种情况。

- HTML内容与其引导的各个部分（CSS/JS/JPEG/GIF)都没有变化：
  - 则向浏览器发送HTTP 304的响应。标示浏览器可以直接使用自身缓存中的信息；
  - 浏览器会从缓存中读取HTML内容；
  - 当HTML开始引导其他部件加载时，浏览器也会从缓存中读取；
  - 此时我们看到浏览器只向Web Accelerator发送了一个HTTP请求，服务器端只是返回了一个100多字节的HTTP 304响应，就完成了整个页面的加载；
  - 加载速度最快，且产生的网络流量也最小。
- 如果HTML内容发生变化，但其他部分未变化：
  - 则向浏览器发送一个HTTP 200的响应，并包含新的HTML内容和适当的缓存控制标签；
  - 浏览器会更新它的缓存；
  - 当HTML开始引导其他部件加载时，浏览器会从本地缓存中读取；
  - 此时浏览器只向Web Accelerator发送了一个HTTP请求，服务器端返回的是一个完整的HTML内容，就完成了整个页面的加载。
- 如果HTML内容没有发生变化，但某些部件发生了变化：
  - 例如：我们修改了网站的Logo (/logo.gif)，此时Web Accelerator会为新的logo.gif生成一个pv代码（可以认为这是一个版本号），并保存到一个新的文件名：/logo.gif;pv=xyz（中间是一个分号），接下来Web Accelerator会把这次请求的HTML对象所包含的logo.gif的连接全都替换为logo.gif;pv=xyz，并更新Web Accelerator保存的HTML拷贝，然后再向浏览器返回HTTP 200，以完成这次对HTML内容的请求；
  - 浏览器会更新它的缓存；
  - 当HTML会开始引导其他部件加载时，除了“/logo.gif;pv=xyz”，其他部件都从浏览器缓存中读取；
  - 同时浏览器会向Web Accelerator请求/logo.gif;pv=xyz的内容，请求成功后这个内容会保存在浏览器的缓存里。对于“/logo.gif”，它仍然存在于浏览器和Web Accelerator的缓存中，但是它以后将不会再被访问，时间久了，缓存回收策略会自动将它从缓存中删除；
  - 此时我们看到浏览器只向Web Accelerator发送了两个HTTP请求，服务器端返回的是完整的一个HTML内容，和我们的新Logo图标，就完成了整个页面的加载。

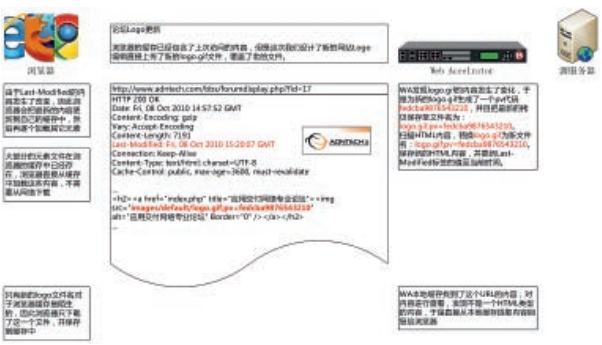


图2　Web Accelerator处理动态内容

这样对于动态内容，原本被关闭的浏览器本地缓存又可以开始工作了，以前我们总是专注在服务器端的缓存实现，而忽略了客户端的缓存控制。Web Accelerator采用的这种方式很巧妙的使浏览器缓存对于动态内容仍然可用。

“反向”动态代理

根据前面的解释，我们可以发现Web Accelerator的部署可能会给原服务器带来额外的负载。所以这里讨论Web Accelerator部署后如何降低服务器的压力。

前面说到Web Accelerator判断内容是否更新，每次都要从原服务器下载HTML，然后进行分析，还要很多次的访问，才能确定。Web Accelerator还有另外一种判断内容更新的机制，就是基于触发器的方式。

下面我们以BBS论坛为例，解释Web Accelerator怎样利用触发器判断内容的更新。

BBS中访问最多的是帖子列表页面和帖子内容页面，那这两个页面什么时候会变化呢？

帖子列表页面（URL：GET /forumdisplay.php?fid=12）什么时候会产生变化？

有人发新帖

```
POST /post.php?action=newthread&fid=12
```

有人回帖

```
POST /post.php?action=reply&fid=12&tid=668687
```

有人修改标题

```
POST /post.php?action=edit&extra=page%3D1&ed
itsubmit=yes
Referer: /post.php?action=edit&fid=12&tid=668687
```

有人访问帖子

```
GET /viewthread.php?tid=668687
```

帖子内容页面（URL：GET /viewthread.php?tid=668687）什么时候会产生变化呢？

有人回帖

```
POST /post.php?action=reply&fid=12&tid=668687
```

有人修改标题或帖子内容

```
POST /post.php?action=edit&extra=page%3D1&ed
itsubmit=yes
Referer: /post.php?action=edit&fid=12&tid=668687
```

我们上面列出了这两个特定页面产生变化的条件，假设只有上面列出的条件才会导致对应的页面产生变化，对于缓存的内容来说只要强制其过期即可。

按照上面列出来的关系，我们定义如下的触发器策略：

每当Web Accelerator收到

```
POST /post.php?action=reply&fid=12&tid=668687
```

或者

```
POST /post.php?action=edit&extra=page%3D1&ed
itsubmit=yes
Referer: /post.php?action=edit&fid=12&tid=668687
```

的时候，就把WA缓存中的GET /viewthread.php?tid=668687设置为内容失效。

只要没有人访问上面的两个地址，GET /viewthread.php?tid=668687就永远都不失效。

Web Accelerator的触发器，就是设定一系列的访问条件，只要匹配，就可以将某些页面强制过期。以这个技术为基础，我们可以让Web Accelerator对某些页面强制缓存，不再每次都要去原服务器进行验证，由此即可降低服务器的压力了。

MultiConnect 技术

浏览器的限制

缺省的情况，浏览器对于一个域名同一时间只能打开两个连接，这样就限制了页面的打开速度。我们都使用过各种下载工具，采用多连接可以更快的完成内容的下载，那如何突破浏览器的限制而不需要让用户做任何的配置变更？Web Accelerator采用MultiConnect技术解决了这个问题。

多域名技术突破限制

一个域名同一时间能打开两个连接，两个域名就是四个，如果有更多域名，那不是成倍地增加并发连接数量了吗？如果我们的域名是www.mctest.com，再增加www01 CNAME www，www02 CNAME www，……这样我们在制作HTML页面的时候随机使用www01， www02，……作为CSS/JS/XML/JPG/GIF的域名，不就可以提高页面的打开速度了吗？

F5 Web Accelerator把这个想法产品化了，你只要告诉Web Accelerator主域名和其关联的后备域名，Web Accelerator就会自动帮你替换HTML页面中各个部件的URL中使用的域名。

使用开源技术实现？

原理讲清楚了，有没有可能利用开源世界的东东搭一个类似的东西呢？也许有这个可能，Squid官方网站上有一个补丁：HTML Prefetching （http://devel.squid-cache.org/projects.html）。这个补丁是用来分析HTML页面并且找到所有这个页面上需要下载的部件，在用户访问这些部件之前提前下载到Squid的缓存里。我觉得这是一个很好的基础，接下来需要做的工作就是生成pv代码了。如果有参与Squid开发的爱好者可以尝试从这个补丁入手。

“程序人生”栏目主持人

周筠，电子工业出版社博文视点（武汉）负责人。带领团队策划出版了《编程之美》、《代码大全》、《程序员修炼之道》、《走出软件作坊》、《我是一只 IT小小鸟》等图书。关注技术的同时，更关注技术背后的个人和团队。

一个“技术文化人”的片段感悟

自2003年加入CSDN后，我的技术身份就很难界定了。曾经有朋友称我为“技术文化人”——不以软件开发为生，但整天都在拿软件开发说事，与这个行业的整体关系可能比任何一个具体的程序员或者架构师都更密切。听上去像是一种恭维，又好像是暗讽，似乎我是站在戏台下面带头起哄的票友。

其实在我看来，我与一线技术人的根本区别，在于关注的问题不同：他们关心的是如何做好软件，我关心的是如何做好软件人。更确切地说，我关心的问题是，对于一个普通的程序员来说，如何通过软件开发这一职业，实现精神上的自由，获得专业上的成就，生活上的安全感，以及对未来的信心。当然，这是很高的目标，在任何社会、任何时代、任何行业，最终都只有一部分人能够到这个境界。但是，我确实曾经把这当成工作的中心，毕竟人的问题才是最根本的问题。

时势可用而不可恃

我接触电脑的时间较晚，一直到1996年，才开始学习电脑操作。最初目标很简单，就是打字、画图、打游戏、看影碟。如果说想写程序，主要就是用来应付一些专业课。但初步学会一点编程后，我意识到编程并不复杂，也就情不自禁地在编程学习上投入越来越多的时间，并最终决定放弃本专业，转行软件开发。对很多和我一样的人来说，放弃自己的专业优势，串行到软件开发领域，“兴趣”固然是一方面原因，而更重要的原因，恐怕还是当时的时代潮流。

在20世纪90年代中后期，随着PC的普及和互联网的出现，国内高校中计算机和软件开发成为显赫一时的潮流，再加上美国克林顿时期的“高科技浪潮”、“新经济繁荣”的光环，以及微软、Borland、金山、江民等成功传奇，使得人们普遍对于计算机和软件行业的未来产生了过于乐观的预期。很多人都觉得，只要搞了软件，成功是唾手可得的。在这种大潮流之下，学校里能摆弄电脑，特别是会写程序的学生，特别受老师器重，在就业市场上也有特别的优势，一个个器宇轩昂、盛气凌人。在这些美梦的诱惑下，“串行”就成了理所当然。

如今，我们这一批“串行生”中，有不少已经成为中国IT产业的骨干，从这个意义上讲，似乎实现了当年的愿望。但IT产业在中国却缩水为一个竞争激烈、外部估值严重下调的普通行业。曾经的美梦，对绝大部分人来说并没有成真。于是很多人在失意之余，也经常产生“悔不当初”的设想。对此我不以为然。

不可否认，当年我们这批人转行IT，有很大的跟风投机成分。对于这个行业及其发展趋势，缺乏基本的了解和积累，对自己的发展也缺少基本的定位和构想，而是看到这个行业的火爆，就迫不及待地想冲进去分一杯羹。进入这个行业后，很多人也继续保持投机心态，今天看到这个火了，就过去捞一把，明天看到那个有上升趋势，就冲过去占位。然而事实证明，我们所处的这个时代，是一个外部环境变动不居、复杂性不断加剧的时代。在这个时代，我们已经不能基于对外部环境的简单预期来制订自己的规划，只有打好基础，积累优势，守时待势。换句话说，时势可以“用”，而不可以“恃”。我认识的成功的技术人，或多或少都经历过一段咬牙坚持的低谷。冲过低谷，就能够获得别人无法企及的积累，时机一

到，便能势如破竹。

抬头看路，埋头赶路

我最初迷上编程，也就是用Turbo C 2.0开发一些DOS下的结构计算和简单的有限元程序，然后用Visual Basic去写一些例子水平的Windows程序，照着书上的例子用汇编语言调用DOS的INT 21h中断。但很快，我就感到不满，我意识到自己对于编程这个领域知之甚少，完全是盲人瞎马，无法确定自己是在朝什么方向前进。在当时环境下，身边没有什么高手可以请教，更没有互联网来开阔视野，我甚至从老师那里得到诸如“DOS将会永远是主流”、“Visual Basic将取代C语言”之类的“专业建议”。这些糟糕的经验让我强烈的不安，所以我决定，在进入这个行业之前，要先对它有一个基本的认识。我的方式，就是大量的、广泛的阅读。

那时候在我出没的范围内，有三四家上规模的计算机专业书店。我几乎每周都要去几次，站在书店的角落里，一读就是半天。书店里一排排的书柜，在我看来就是了解软件开发这个行业的一幅幅地图，不管是有关的、无关的，看得懂的、看不懂的，听说过的、没听说过的，我都不放过。通过如饥似渴的阅读，我了解到，除了DOS和Windows 95之外，世界上还有Windows NT和UNIX，了解到Win32不是Windows 3.2，COM跟.com不是一回事，VBA也不是Visual Basic Advanced，了解到Delphi正在跟Visual Basic激烈竞争，了解到C/S体系结构的含义，也明白了当时仍然走红的FoxPro将很快被SQL数据库所取代。逐渐的，我的大脑里出现了一幅软件开发领域的全景地图，尽管在今天看来，这幅地图非常不精确，也并不全面，但是对当时的我来说，已经可以用它来为自己的学习导航了。

事实上，这段时间的经历对我正反两方面的影响都非常深远，一方面，我由此形成了对软件开发领域的全局性的理解；另一方面，过于关注大格局，使我少了埋头钻研的恒心，对关键领域深入不够，这又成为我的遗憾。

我后来在CSDN工作的时候，曾经用“抬头看路”和“埋头赶路”这两个状态来描述一个程序员理想的学习周期。“抬头看路”，就是专门拿出一个时间段，把所关注行业的大趋势看清楚，并结合自己的情况，设定目标和计划。“埋头赶路”，就是在目标和计划设定清晰之后的一段时间里，把自己封闭起来，“两耳不闻窗外事”，不再关心行业的风云纷扰，而是踏踏实实实现自己的目标，形成特长。

以我来说，我那时拒绝了计算机老师主攻Visual Basic的建议，果断选择C语言作为自己的主攻方向，应该说是基于“抬头看路”所得出的正确决策。而之后过早地从C过渡到C++，则应该说犯了一个错误。C语言的小巧、明快、圆满和强大，迄今无出其右。由于其语言简捷，没什么可学的，学习者的旺盛精力将很快“被迫”转向真正有价值的东西——算法、数据结构、编译、图形、系统编程，等等。我后来认识的很多高手，就是因为早走了几步，“没听说C++”，就在C上下了苦工夫，“埋头赶路”，反而“因祸得福”练成了很强的动手能力，而能有一方成就。到后来我在CSDN工作的时候，这方面的体会就更深。那几年里，为了能够与各路高手平等交流，我几乎涉猎了所有重要的技术领域，研究了大多数热门的技术概念，阅读之广，尝试之杂，远超过一般软件开发者的需要。正因这种“博”，使我对各技术派别及各主要企业间的关系和沿革了然于心，从而对于行业发展形成自己的见解。但俗话说“样样皆通，必然样样稀松”，广泛涉猎的代价就是深入不够，对此我可谓有切身之痛。

遇高人不可交臂而失之

我在初步掌握C语言之后不久，就一步踏进C++。C++复杂的语法、强大多样的抽象机制、奇妙的各种语言现象，极大地满足了我的好奇心和求知欲。但是C++的真正挑战在于从“知”到“行”。一次计算机图形学大作业受挫的经历，让我强烈地意识到我的C++水平其实非常低下，于是就到处寻找能够提高C++水平的书。很自然的，MFC就被我纳入视线之中。

当时正值MFC处于其顶点，所以我买了好几本大部头的MFC书来啃。但越是使用MFC，我就越是不满。我完全无法把MFC与我熟悉的C++连接起来，两者之间似乎存在一个巨大的断层，让我觉得MFC完全是另一门语言。我花了不少心思去猜测、分析，并试图阅读MFC的源代码，但是那时候的能力还非常浅薄，完全无法理解MFC的奥秘，也无法弥合那个断层。很多次，我都不禁灰心地想，也许自己并不是写程序的料，或者不是搞C++的料，或许应该放弃。

就在我最为困惑和郁闷的时候，一天下午，我在书店柜台上发现了一本名为《深入浅出Windows MFC程序设计》的新书，作者侯俊杰。我只站在那里翻了五分钟，就被其中的内容“雷”得头皮发炸——这本书不但正中靶心地打到我的兴趣点上，而且语言之优美，内容布局之巧妙，都是前所未见。记得那本书的价格不菲，我当时负担不起，于是找同学借钱买

下来。接下来几个星期，我每天捧着这本书反复琢磨到深夜，感觉所获得的长进，比前面几年都要多。通过这本书我了解了一个完全超过我之前层面的C++的世界，也把作者的名字牢牢记住。几年后，这本书的第二版以《深入浅出MFC》为名发行，畅销海内。2000年底，我动手翻译了STL之父Alex Stepanov的一个长篇访谈，因为这篇文章，我得以认识CSDN的掌门人蒋涛，并且经他介绍，终于认识了心仪已久的侯先生。

与侯先生的相识，是我C++学习生涯的一个重要转折点。当时侯先生也已经将注意力转向泛型和STL技术，并且在《程序员》杂志上发表了著名的《C++大系》系列文章，为国内的C++学习者打开一片全新天地。与侯先生刚刚认识不久，他就通过当时《深入浅出MFC》的编辑周筠女士，向我赠送了好几本“C++大系”中的重要作品。我至今还能回忆起打开大包裹时兴奋得几乎要晕厥过去的感觉，也清晰记得那其中的内容：Scott Meyers的Effective C++和More Effective C++，Bjarne Stroustrup的The C++ Programming Language，Matt Austern的Generic Programming and the STL，Nicolai Josuttis的The C++ Standard Library和那时刚刚出版的Andrei Alexandrescu成名作Modern C++ Design。一下子有这么多经典，我几乎废寝忘食地阅读、试验，遇到困难，就用邮件向侯先生请教，在侯先生的指导下，仅仅几个月，我对C++的理解便上了一个大台阶。特别是Scott Meyers的两本书，对我的作用可以说是醍醐灌顶、脱胎换骨。在那之后，侯先生又邀请我与他合译The C++ Standard Library，以共同合作的方式给我另一个层面的教益。之后几年里，侯先生总会时不时地给我帮助和关怀，帮我购买珍贵的资料，关心我的事业和生活。与侯先生的交往，可以归为八个字：知遇之恩，师生之情。这段回忆对我来说，是深藏心底的一份温暖，也是一份歉疚。侯先生曾寄希望我走入技术写译和培训的行业，并几次为我创造这样的机会，但我却最终没有从命。虽然多为国内现实所限，却也少了报答先生的机会。

对于一个学习者来说，高人的点拨确实可令人“顿悟”，走到一个全新境界。

实际上侯先生不光帮助了我一个人，在七八年前，我们这群在内地的C++爱好者中，先后直接受到侯先生帮助的有数十人，其中有不少与我保持联系。据我的观察，这些人中的大部分都有着还算不错的发展。如果算上侯先生图书的读者，侯先生帮助的人何止十万众。一个人的贡献，可至于斯！

所以每当我总结这段历史，不禁心生感叹，对于一个学习者来说，高人的点拨确实可令人“顿悟”，走到一个全新境界。因此，遇高人不可交臂而失之。我也知道，像侯先生这样品质和才情的高人毕竟极少，但我认识的大部分技术高手，其实都是可师之人。如果能够放下身段，认认真真向高人请教，那么对于自己的成长和发展，都将有莫大的好处。

文武之道

经过多年观察和思考，我认为我发现了程序员实现事业发展的一个关键原则，那就是在编程技术上保守一点，而在专业及行业领域进取一点。我称之为“技术组合的文武之道”。有趣的是，这个“文武之道”，恰好与一般程序员的实践相反——大部分程序员在编程技术上比较激进，却疏于在行业领域下工夫。

最早注意到这个问题，是在研究生实习期间。那时我被导师派往清华同方参与一个大型专业软件项目的开发，体验到了真实环境下的团队软件项目开发。这个项目由于脱离具体的条件，目标过高，与同期很多雄心勃勃的科幻项目一样，最终以失败告终，但是这段经历却使我受益良多。正是在这个专业项目的开发当中，我重新认识了领域知识与编程技术之间的主次关系。

项目本身的目标是开发建筑工程企业的ERP系统，第一阶段的任务是形成概预算自动计算功能，其中涉及大量的图形绘制、对象建模、数据存储等我非常感兴趣的技术内容。在参与这个项目的初期，我非常兴奋地设想过一个雄心勃勃的技术方案：用VC/MFC为开发工具，自主开发图形库和建筑构件类库，支持3D可视化建筑建模，并且用类似对象数据库的方式进行持久化，等等。然而真正进入这个项目，我发现项目主管并没有带领我们冲向这些令人兴奋不已的技术高峰，而是一个会接着一个会地分析需求，研究概预算规范，了解有关领域知识。这个冗长烦琐的过程，让我大失所望。然而更郁闷的是，当项目进入到编码阶段时，项目主管竟然选择Visual Basic作为开发工具，而数据库服务器更是非常阳春的Jet DB，也就是Microsoft Access。我对此强烈不满，几次酝酿之后，终于找到项目经理，

反映我的想法。这位项目经理是清华的毕业生，已经通过Visual C++的MCSD认证，是当时极为罕见的技术人才。他陈述了自己的看法：这个项目是一个专业软件，主要的困难和障碍都集中在专业领域，而在编程技术本身，无论是VB还是Jet DB，都足以在现阶段满足要求。在这样的情况下，如果选择MFC或者其他更酷的技术，无疑会大大增加技术实现的难度，纯属是毫无意义的自找麻烦。他的原则，就是尽可能用可实现项目需求的最简单的技术来完成任务，因为技术越简单，项目风险就越低，团队开发的管理成本就越小。

事实上，当年我骄气很盛，并没有被说服，反而觉得他缺少冒险精神。但是他对于编程技术与领域知识间轻重关系的阐述，确实也引起我的思考。在项目实践中我的确发现，相对于变化多端的概预算规则，在屏幕上画个3D的房子并不是这个项目的关键。在一起工作的专业程序员们，似乎很快就可以完成类似这样的任务，但是面对复杂的业务规则，他们就莫名其妙，非常无助了。

毕业后，我的第一份工作是在联想做掌上设备的软件开发，在那里我也观察到相同的问题。在联想，我搞了一些编程技术上的创新，把SGI STL和Boost的几个组件移植到Windows CE平台上，并自己写了一个远程设备的软件调试库，还为几个自主软件设计了非常符合设计模式精神的架构。但是所有这一切都没有得到大家的认可，大家关注的中心，是新设备的产品特色、软硬件配置、营销重点、成本控制、项目实施等等。领导和同事们对于我的这些努力，既没有鼓励，也没有反对，而是任我自便，我当时对此确实是大有“怀才不遇”的抱怨。但几年之后，当我有了更多的职业积累、更广阔的视野和更成熟的心态之后，我才能更客观地反思自己在联想期间的经历。其实，无论是手机、掌上电脑还是平板电脑，消费电子类产品的开发当然是以产品设计、营销规划和工程控制为中心，这才是这个行业的关键领域知识。相比之下，什么STL、设计模式，完全不是重点。如果我当时的心态更成熟一些，能够主动学习和掌握消费电子产品的产品设计专业知识，让自己的编程技术能够为组织的整体目标服务，那么我今天的职业发展可能会是完全不同的另一番模样。

进入CSDN之后，我的所见所闻就更多了。大量的技术高手，把主要精力放在“改善程序员开发体验”的事情上，比如新潮酷派的语言、漂亮的开发环境、方便的代码生成和魔术般的设计抽象，倒是能赢得程序员圈子里的一片叫好声，但是却得不到客户的认可。反而是很多团队，由于对应用理解到位，用普通的技术为客户创造了巨大的价值。比如我认识的一家企业，经过深入交流和分析之后，发现移动财务审批是客户没有提出来、但却非常需要的功能，于是用简单可靠的技术手段，实现了这个功能。客户非常兴奋，大加赞赏，甚至破天荒地将其在全国各地的中层干部召集到北京，专门针对这个功能进行培训。

这些经历和见闻，逐渐使我意识到了领域知识与编程技术之间的主次关系。事实上，从客户和管理者的角度来看，你使用什么先进的编程技术并不重要，重要的是你的软件能够正确地做好该做的事情，你是否能对所解决的现实问题有深刻的认识，有所洞见，有所创新。很多时候，在程序员圈子里被追捧的新概念、新思想和前卫技术，实际上并没有经过时间和实践的检验，存在很多问题，过于积极地引入它们，对于客户的利益，反而是一种危害。真正能够意识到这个问题的开发者，在编程技术本身上会趋向于稳妥保守，不急于追新赶潮，宁可做一个后知后觉分子。但在行业领域，他们则完全是是另一个姿态，积极进取，敢于创新和尝试，总是殚精竭虑地思考如何为客户提供更合用的功能、更高的价值。在我所见范围之内，凡是这样做的，不论是企业还是个人，也无论是在企业应用市场还是在消费类市场中，都能够更快地走向成功。

正是基于这个认识，我在很多场合都提出程序员要懂行业，或者重视领域知识。我相信，如果能够掌握好编程技术和行业知识之间的文武火候，作为程序员个体，无论是在企业中服务，还是自己创业，成功会相对容易些。

结语

2009年，我开始了一段新的人生尝试，究竟将走向何方，尚未可知。但对于我来说，作为“软件文化人”的这段经历，已成为我生命中的一段充满精彩与遗憾的篇章。我在这里用了这样一种松散的手法，随意收集了几个片段和感悟，由于漏掉了许许多多帮助过我的人，这篇东西是不可以称为“成长故事”的。但写下来的这几个散片，倒也确实是我有感而忆，有感而发。不知道过几年以后，再读到这些，是否又会有不同的看法？那就不得而知了，到那时再说吧。

作者简介

孟岩，现就职于IBM中国公司企划传播部。曾任CSDN和《程序员》杂志技术总编。

■ 责任编辑：董世晓（dongsx@csdn.net）

这个小人不简单

■ 文 / 潘加宇

通过可乐、投注站、喝水等几个生动的实例，展示了理清软件功能主要为谁服务的重要性。随后得出了结论：需求要具体，设计要抽象。或者说，需求，要把产品当项目做；设计，要把项目当产品做。

执行者（Actor）的意义

一个老头找到PS可乐公司，告诉他们的主管说：“我可是你们的忠诚客户啊！我喝过的可乐罐排成线，可以从苹果园排到通州。现在我老了，我对你们的可乐下一个版本提出如下需求：第一，我有胃病，下一个版本不要放碳酸气；第二，我有糖尿病，下一个版本不要放糖。”PS可乐公司的主管很感动，哇，这么棒的顾客，把需求提得那么具体，省去我们调研需求的好多时间，好，下个版本就这么办！

会有这样的场景吗？不会的。老头老太太可以买可乐喝，甚至可以买给自己的狗喝，PS可乐公司不会拦着。问题在于，老头老太太提的要求，或者他们的狗提的要求，PS可乐公司不会放在重要的位置来考虑，因为可乐的目标客户是年轻人。可惜，很多时候我问开发人员：“可乐卖给谁？”得到的回答大多是“卖给消费者”、“卖给想喝可乐的人”——对做出好卖的可乐没有帮助的、正确而无用的废话。

我啰嗦这么一大堆，其实就是想说，从“可乐可以喝”到“年轻人喝可乐”，多出来的这个“人”背后的意义不简单。

软件系统也一样。如图1所示，需求的表达方式，从“系统提供定义推广规则的功能”变成“投注站老板使用系统来定义推广规则”，重要的意义就在于发现了执行者（Actor）这个小人。电脑开着，投注站老板上洗手间去了，有人路过也可以使用投注站系统来定义推广规则，但“定义推广规则”这个用例所包含的各种功能需求和非功能需求，更看重的是投注站老板的意见，而不是路人的意见。

和之前的需求技术相比，用例技术（包括执行者和用例）的不同之处就在于发现了“卖”——需求是研究软件怎样好卖的技能。以前有观点以为执行者映射了权限管理，意味着系统需要有权限控制，其实这是一种误解。一罐可乐打开在那里，猪路过也可以喝，可乐本身并没有“防猪”的权限管理，但猪仍然不是执行者，因为猪不是可乐的目标客户，PS可乐公司不会因为猪喝得不爽就改变可乐的功能和性能。当然，如果你做的是这样一种“可乐”，猪喝了可乐可以长得更快，那就是另一回事了。不过，你能想到“猪喝的可乐”，不也是拜了执行者思维之赐吗？

从所有人到一群人

为什么卖给谁这么重要？因为竞争使得产品分类越来越细，不再有针对所有人的产品了。在原始人眼里，喝水是很简单的事情，靠着河就喝河水，靠着泉就喝泉水。随着市场的发展，喝水变得越来越复杂，如图2所示。

软件的分化也是如此，见图3。程序员想用QQ完全可以用，没人拦着，但他们的意见对QQ的需求影响有多大呢？

所以，通过思考执行者来理清楚“这个功能主要是为谁服务”非常重要。不过，思考出这个小人不容易，特别是当要

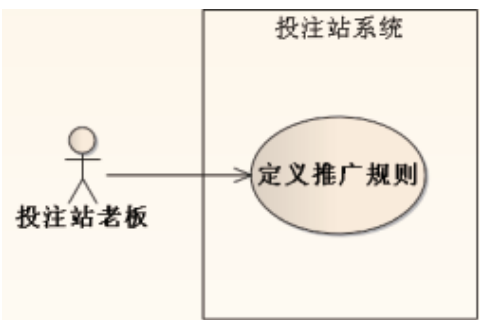


图1 用例和执行者：卖给谁和卖什么

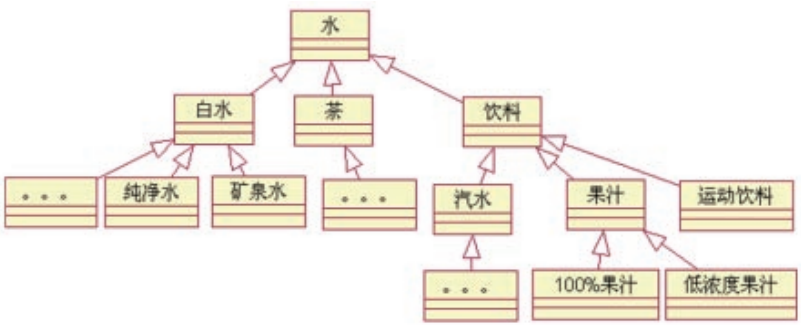


图2 不断分化的水，每一种水针对不同的人群

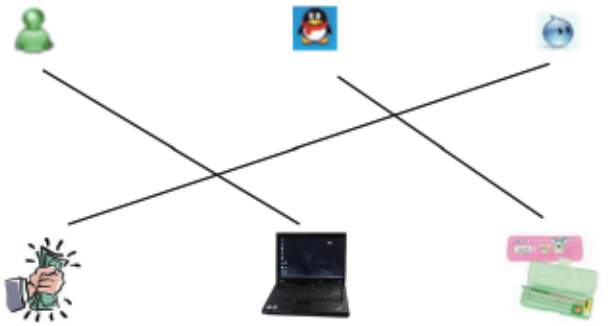


图3　软件的分化，MSN、QQ、旺旺针对不同的人群

开发的系统所服务的组织不是一个有明显岗位分工的正式组织（例如国土资源局），而是一个松散人群（例如玉米、房奴、驴友）的时候。经常有开发人员问：如果只是做一个网站展示企业和产品信息，要不要建模呢？确实，象完成作业一样做一个展示企业和产品信息的网站容易，可是，网站要做到能给企业带来利润，企业下次还找你，那就需要做艰苦的调研和建模。光是“网站给谁看”这个问题，就已经使很多人栽倒了，给“春哥”和给“著姐”看的网站能一样吗？

给执行者命名的时候，要拒绝淡而无味的命名，以及正确而无用的废话。看看对执行者的命名，就可以看出开发人员是否具备探索需求所需的市场思维。有时为了想这个小人的名字，要绞尽脑汁，这样的需求才有味道、有价值。

起淡而无味的名字，原因除了开发人员对涉众缺少调研之外，其他原因也可能是开发人员忍不住“透过现象看本质”——“单证人员”等各种人员都可以从“用户”继承下来，这犯了从设计的角度找需求的错误。设计时可以用“用户”，这样许多特征都可以复用，但做需求时是不考虑复用的。

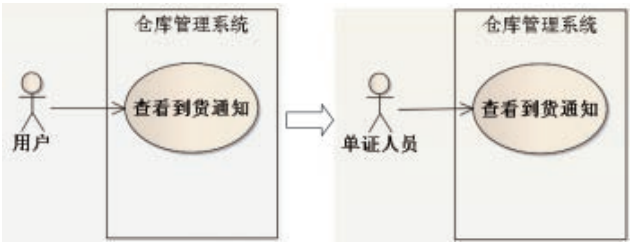


图4　拒绝淡而无味的命名

我们去小吃摊上买馄饨，老板给我们饺子或锅贴，还辩解说“都是面和菜肉的组合”，这样可以吗？我在之前的文章中说过：利润＝需求－设计。小吃摊老板的赚钱之道就是用少量的材料（设计组件）灵活组合出不同的需求（用例）卖给不同的顾客。

需求要具体，设计要抽象。或者说，需求，要把产品当项目做；设计，要把项目当产品做。

从一群人到一个人

既然需求要具体，定位到目标人群后，有时甚至还要定位到一个人，假装这个用例专门为他而做。

例如，某个功能为男性提供，那么如何调研详细的需求？是对男性群体做大规模调查，还是从CSDN员工里随机抽取一位男性来深入访谈、观察？一种方法是问：哪个男人最像男人？得到具体的人物“春哥”，然后，假装这个用例就是专门为春哥而做，这就是交互设计中用到的Persona方法。

如果一下子想不出来哪个最像，可以通过比较，慢慢逼近答案。例如，说“客户是医院”还不够，还要问，“协和医院”还是“大兴中医院”更像你的系统所服务的医院？

做这样的思考是很有意思的：哪一种水果最像水果？橙还是枣？从科学的角度说这两种水果是平等的，但我们说到水果时，想到更多的是橙还是枣？哪一种方便面最像方便面？哪一种CT机最像CT机？哪一个SNS网站最像SNS网站？哪一个程序员最像程序员？

目前，介绍GoF23个设计模式的书非常多。大多数书籍在讲述时按照创建型、行为型、结构型的顺序进行。在这23个模式中，有一个模式经常会放在前言先行介绍，作为例子展示设计模式的魅力，您知道是哪一个吗？也就是说，哪一个设计模式最像设计模式？

从台上人到台下人

再继续往下探讨，执行者的原意是“演员”，演员在台上如何表演，由台下各类观众的口味综合决定。软件的需求也是一样，由各类涉众的利益综合决定，通过执行者和系统的交互演绎出来。将执行者和涉众分开之后，涉众一定是人，执行者可以不是人，也就是说，银幕上演的是喜羊羊和灰太狼，也同样要考虑观众的口味，而用例之前的需求技术所使用的“用户”这个词，相当于把演员和观众混在一起了。关于执行者、涉众和用例的进一步探讨，留待下文“这个圈圈不简单”。

作者简介

潘加宇，UMLChina首席专家。1999年创建UMLChina，潜心研究需求和设计技能。2002年开始对外提供UML需求和设计的技术指导和训练服务。

■ 责任编辑：董世晓（dongsx@csdn.net）

适度设计

■ 文 / 杜玄 魏崴

如何平衡软件开发中设计的“度”是困扰众多技术人员的问题。本文给出的适度设计的真谛是：紧扣用户需求，不要猜测未来，够用的设计就是好的设计。

何谓“适度设计”？这个概念是相对于“过度设计”所提出来的。俗话说“水满则溢，日满则亏，月圆则缺，器满则倾”， 对于软件设计来说，也要有一个度。十几年前，软件开发刚刚起步的时候，软件开发界中普遍存在的问题是：没有任何设计而直接写代码。有些人还美其名曰“代码即设计”。至少在国内，这种直接写编码、到处拷贝代码、代码写到哪里就设计到哪里的现象，在当时是普遍存在的。那个时代没有“过度设计”的问题，那时的问题是没有任何设计，也没有设计模式的概念，开发项目就是直接的代码堆砌。从那个年代过来的软件开发者，翻翻当年的软件项目，大多存在这样的毛病。

然而，今天软件设计的天平却向相反的方向倾斜了。随着时代的发展，人们越来越认识到设计的重要性，认识到设计原则、设计模式的重要性。人们提出了低耦合、高内聚、开闭原则等软件设计理念。在现在的软件项目中，大量应用着这些设计理念。为了解耦，为未来的变更预留足够的灵活性，大量使用设计模式和接口隔离、配置文件隔离和动态加载等技术。首先，我要说，我认同这些设计理念。但是，在自己拥有源代码的、可控的软件项目内部，过多的、过早的应用，成为“为了用设计模式”而应用设计模式的地步，就有些过头了。忽视了软件设计的根本是满足需求，盲目追求所谓的“先进设计”，人为地把一件简单的事情复杂化，这是对这些理念的滥用。

我们需要的是软件设计天平的平衡，既不能没有设计，也不能“过度设计”。“适度设计”应该是我们今天所关注的事情。

下面讲一个虚构的软件开发的故事。虽然是虚构的，但是故事中的很多情节都是我们在实际项目中所遇到的。从这个故事中我们可以看到一个软件项目的演变过程：开发者由于对未来变更的臆想和对设计原则和设计模式的滥用，而导致“过度设计”。一个本来如此简单的的需求，却用了如此复杂的设计来实现。

故事是这样的，一家生产汽车的公司提出了一个软件需求——需要一个汽车产品生产和管理的软件，软件负责组装和测试汽车。这家公司很小，只有一个生产工厂，也只生产一种型号的汽车。为了叙述方便我们把这个提出需求的公司叫做公司A。

一家软件开发公司接到了这个软件开发项目（简称为公司B）。公司B找来一个软件开发人员小刘具体负责设计开发这个项目。小刘入行不久，只有面向过程的软件开发经验，面向对象的知识不多。小刘分析完用户需求，准备用C#来实现这个软件项目。他根据需求写出了以下的代码：

```
using System;

class Car...

class Program
{
    static Car car;
    public static void assembleCar() {
        car = new Car();
        //do assemble
    }

    public static void testCar(){
        //do test
    }

    public static void Main(string[] args) {
        assembleCar();
        testCar();
    }
}
```

由于公司A很小，所提出的需求也很简单。如此简单的需求，直接写代码也不会有什么不清楚。上面这段代码很短，但是完全满足公司A的需求。开发人员用了两个函数，一个是assembleCar()用于装配汽车，另一个是testCar()用于测试汽车。这两个函数按着顺序依次各调用一次，汽车就生产完成了。项目确实如此简单，小刘虽然用面向对象的C#语言做了一个面向过程的实现，但他也只用了一天就完成了任务。

这时，公司B的资深软件开发人员老王看到了这个项目的设计。老王最近正在研究面向对象的软件开发，他对小刘说：“为了软件将来的灵活性和可维护性，我们应该用面向对象的技术来实现这个项目，不能简单地用过程函数实现，这样无法发挥C#语言面向对象的优越性。”老王说干就干，他用UML进行了设计，并且实现了原型代码。老王画UML，写代码，花费了两天时间做完这个软件项目。

首先，老王用UML定义了主程序类Program、工厂类Factory和汽车类Car。在工厂类中定义了造汽车的方法makeCar()。在汽车类中定义了装配汽车的方法assembleCar()和测试汽车的方法testCar()。Program类中要引用Factory和Car类，所以Program类要依赖它们两个类。同理Factory类依赖Car类。如图1所示。

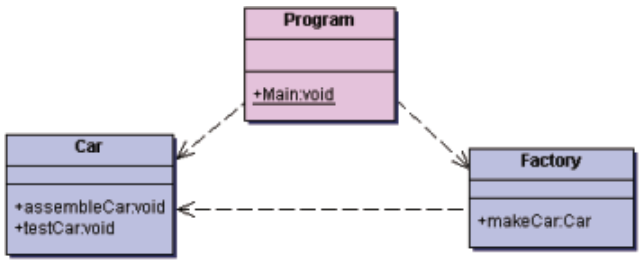


图1　老王使用UML做的设计

代码实现如图2所示。

这个代码实现也忠实地完成了用户需求的要求，同时用面向对象完成了软件设计。

小刘看了老王的设计实现连声叫好，设计中有汽车类、工厂类，分类清晰明了，比面向过程的代码好多了。

正在小刘称赞老王的设计的时候，公司B中的技术“牛人”来了。这位“牛人”读过很多关于现代的软件开发设计的书籍，对高耦合、低内聚、设计模式和开闭原则等非常认同，平时一谈起软件设计，满嘴都是架构、模式和解耦什么的。 他一直想把这些设计理念用在实际的软件项目中去，只是苦于没有机会。正巧“牛人”看到老王的设计，非常地不屑一顾。他说：“Program、 Factory和Car三个类互相依赖，它们紧密地耦合在了一起，这样的设计很差，一定要解

```
using System;

class Car{
    public void assembleCar() {
    }
    public void testCar() {
    }
}

class CarFactory{
    public Car makeCar(){
        return new Car();
    }
}

class Program {
    public static void Main(string[] args) {
        CarFactory factory = new CarFactory();
        Car car = factory.makeCar();
        car.assembleCar();
        car.testCar();
    }
}
```

图2　老王设计的代码

耦，让我给你看看什么是灵活的架构设计”。

“牛人”说，公司A现在只生产一种车型，而将来它们可能还生产其他型号的汽车，所以软件中不能直接定义Car这个类，而要把Car抽象出来，要用一个接口类ICar隔离实际的Car。虽然现在只有一种类型，但是软件中先预置一个ICar，为以后增加可扩展性。“牛人”又说，公司A现在只有一个生产厂，将来也许可能有多个生产厂来生产汽车，如果在软件中直接写Factory类，以后修改会比较麻烦。所以决定再用一个IFactory接口来预留未来多个工厂的情况。“牛人”进一步分析，虽然Car和Factory都用接口隔离了，但是Program中还是要实例化具体的Car类，并且还要判断不同的具体的Car在不同的生产厂生产。这样在Program中还是对具体Car和具体的Factory有强依赖。“牛人”又有了主意，现在流行配置文件和动态加载，可以把Program中的耦合解耦到配置文件中，这样在Program的代码中就只出现对ICar和IFactory的依赖，再也不用依赖具体的实现类了。基于这套想法，“牛人”花了三天的时间完成了这个项目的设计。如图3所示。

如图4中代码片段①~③所示。在代码片段①中，CarA和CarFactoryA虽然只有一个，但是已经用ICar和IFactory做了隔离。CarFactoryA被封装在了一个动态链接库DLL中，如代码片段②所示。Config.ini（见③）是动态加载的配置文件，通过配置Car与DLL的关系，可以自由地配置不同型号的Car在相应的CarFactory生产厂生产。

“牛人”的设计堪称完美，预知了各种未来变化的可

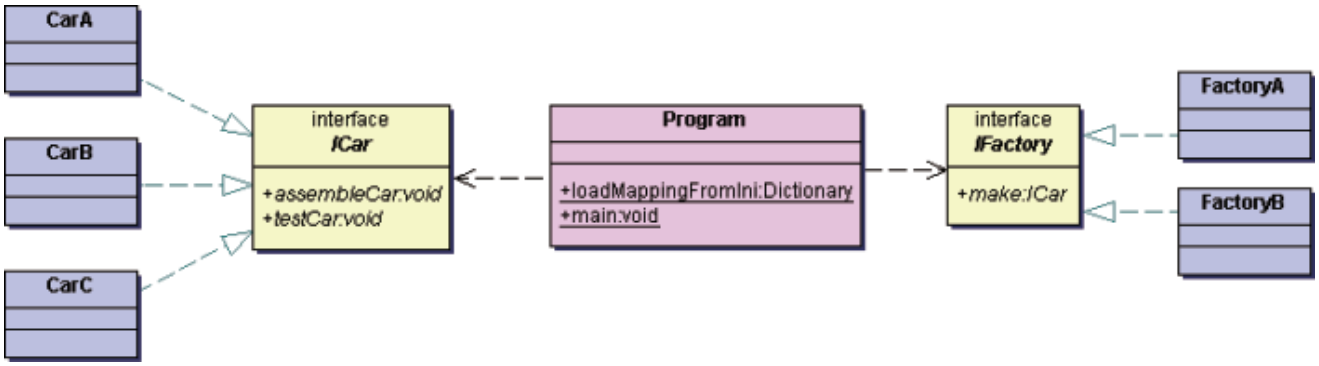


图3 “牛人”的设计

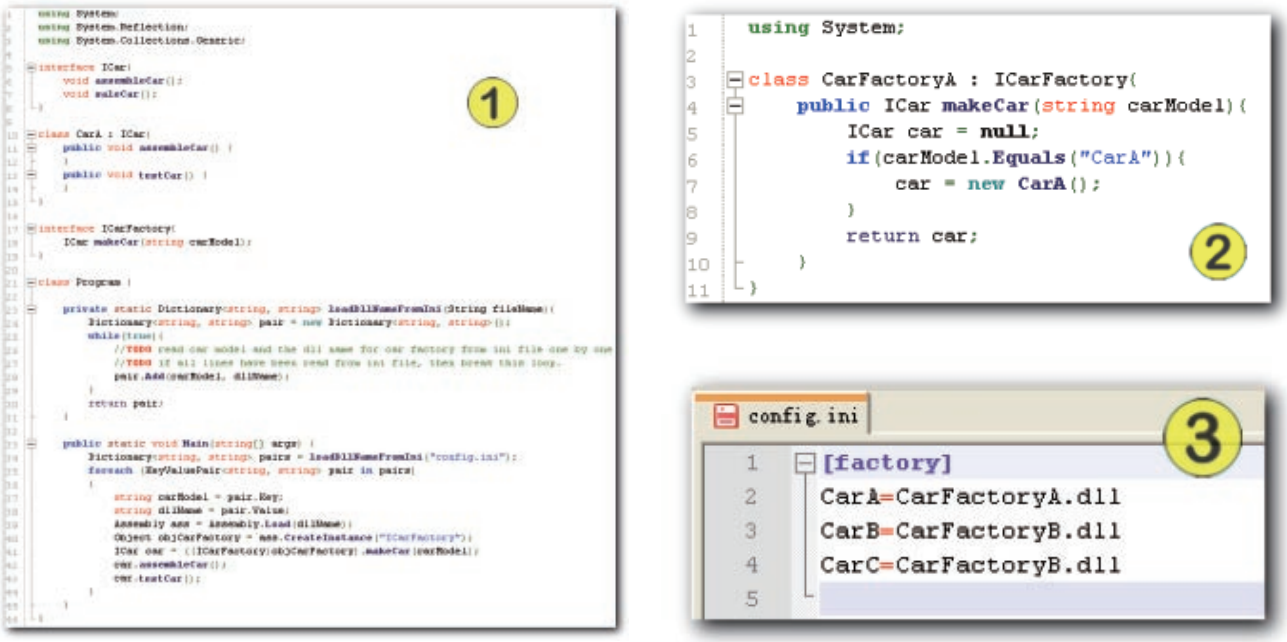


图4 代码片段①~③

能。小刘和老王都非常佩服这个“完美”的设计。不过，小刘和老王开发设计用的时间较短，而“牛人”用了较长的开发设计时间。“牛人”说：“磨刀不误砍柴工，将来你们就知道我这个设计的好处了。”

在软件交付给公司B后不久，公司B提出了新的需求，工厂不但要装配汽车和测试汽车，还要设计汽车。

如果是小刘的方案，增加一个designCar函数就搞定了。老王的方案在Car类中加一个designCar方法也搞定了。“牛人”的设计有点麻烦，由于用了接口隔离，需要先修改接口ICar，才能修改实现类Car。

问题在哪儿呢？小刘和老王的方案面对变化时很简单地一步就解决了，而“牛人”的设计却需要两步才能完成。“牛人”想象了很多未来的可能需求，但未来并不是按他的想象来发生。他对多种车型号和多个工厂的情况花了很多精力去预制了灵活的框架，而实际的需求是汽车本身工作的相关变化。对此，“牛人”的框架没有办法，只好增加接口方法来应对。

对比三人的设计，小刘的设计太落伍，“牛人”的设计是过度设计，老王的设计对现有需求来说，可能正好合适。

还好，这是一个虚构的简单故事。当我们面对一个复杂系统，在过度设计的框架上已经做了很多实现，而有一天精心搭建的框架却不能应对新的需求时，修改起来比适度设计的要复杂得多。无论你如何想为未来做多么充分的准备，也永远是不充分的。因为你不是上帝，你不可能预知未来。所以，软件设计“适度”即可，重要的是要迎接变化，快速地

随需而变，这才能保持软件系统的生命力。

怎样才能快速地迎接变化呢？不要忘记，现代的软件工具与10年前相比已经进步很多，工具的代码重构能力也已经非常强大。面对现有的需求，我们做出一个刚刚满足现有需求的软件设计，当需求变更时，用重构工具辅助重构代码以适应新的需求。老王的软件设计，对当前需求，就是简简单单的三个类。当有新需求的时候，比如用户要生产第二种类型的汽车时，可以用重构工具抽取ICar接口，以适应多种车型的需求。图5是C#的免费开发工具SharpDevelop抽取接口的界面截图，对于其他语言也都用相应的工具，比如Java有Eclipse这样强大的工具支持代码的重构。

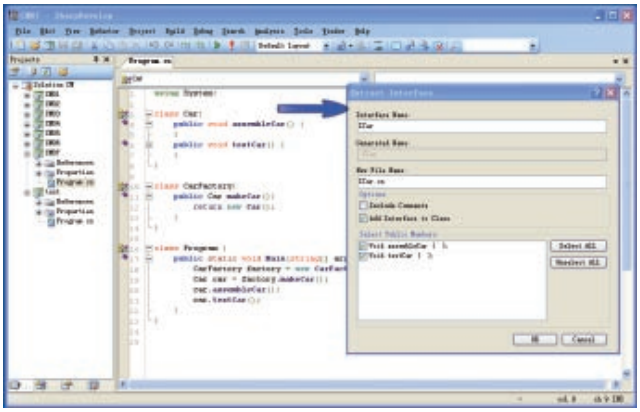


图5　SharpDevelop抽取接口的界面截图

现代的开发工具重构功能很强大。比如可以给对象、方法等改名，工具会自动把引用的地方全部自动改过来，还可以抽取方法、函数上移、抽取接口等。由于自动重构工具是代码不同形式的等价变化，重构并不改变其原有的功能，这样就为我们整理代码、变更代码提供了极大的方便。

适度设计，积极重构，使代码紧贴现有的需求，使系统紧凑精悍。

下面提出几点需注意的要点，以此来遵循设计的平衡。

- **不要追求“零耦合”，而要合适的耦合。**我们在软件设计中一直追求的是“解耦”，希望软件模块像建筑中的预制件一样的标准化、通用化，这样软件就可以像建筑预制件一样并行化地生产，从而提高软件开发效率。我们要记住一点，追求“解耦”，并不是追求“零耦合”，因“零耦合”是不可能实现的。“零耦合”的两个模块根本就不是需要放在一起的模块，并且软件设计也不能追求“无限接近零耦合”，追求“无限接近零耦合”的人就是最容易犯过度设计毛病的人。为了追求“零耦合”，凭想象预留过多的接口，导致接口和配置文件的灾难。适度设计提倡的是适度耦合，按需解耦，只有存在解耦需求的时候才通过重构现有代码实现解耦。
- **不能只知道开闭原则，也要知道“易重构原则”。**开闭原则是对的。这里提出“易重构原则”，是要使软件系统不能在细粒度上滥用隔离，尤其是配置文件隔离，过度的滥用将导致接口和配置文件的灾难，细粒度上的滥用结果将造成系统难以重构。好多系统接口过多，很多细小的模块也用动态的方法通过XML实现隔离，这样的系统，已经被XML文件隔离得支离破碎，无论是通过人，还是重构工具都非常难以实现代码重构。如果人工重构，常常不如重写代码来得快捷。
- **不要引入第三方来实现解耦，要尽量保持代码的纯洁和单一。**对于跨语言的重构工具还没有出现，在现有的技术条件下，如果一个系统被各种配置文件切割开来，想大规模重构是很困难的一件事。不如尽量保持用一种语言解决问题，尽量保持单一、纯净。
- **不要过早引入框架和设计模式，够用的框架就是好框架。**过早引用复杂框架是自己找麻烦，有精力不如好好重构一下旧代码。
- **不要封装旧代码，而要重构旧代码。**不能因为不敢碰旧代码，就简单封装一层了事，而是要积极地重构代码，保持旧代码的健康。这里说的是自己有代码的情况，可以引用别人的DLL；没有代码另当别论，这种情况只好用自己的代码封装了。
- **项目管理中缺乏代码重构阶段。**很多问题的根源在于，软件项目管理中，根本就没有给代码重构的时间。尤其在项目的维护阶段，只会修修补补，就像建筑根本没有准备大修的时间。

最后，对于软件研发，我们要保持设计“度”的平衡。紧扣用户需求，不要猜测未来，够用的设计就是好的设计。面对需求的变更，要积极重构，这样才能保持代码的健康和活力，才能保证软件产品的及时发布。

作者简介

杜玄，就职于泰乐（Tellabs）通讯，从事网管软件的开发和研究。

魏崴，就职于泰乐（Tellabs）通讯，从事网络管理系统的研究开发工作。

■ 责任编辑：董世晓（dongsx@csdn.net）

美丽邂逅

——解疑用户体验设计

■ 文 / 谢旭鸿

同乘班车的路上，作者与同事关于用户体验设计的精彩问答。

搭往公司的班车，遇到一个其他部门的同事，他问的很多问题，引出了本文。这些问题，其实我也经常被其他人问到，这其中，既有我们亲密合作的伙伴，如产品经理、开发工程师（程序员），也有对于产品决策有绝对发言权的老板，还有平时和我们工作交集不多、但却是一个成功的产品生态链上重要的一环的角色，比如来自客服部门和销售部门的同事。所以做些加工，整理成文，希望有助于大家了解用户体验设计部（简称 UED）在网站开发流程中的角色和作用，从而有助于不同部门的协作和配合。

Tom：请问你是什么部门的？

Heidi：我是国际站用户体验设计部的，就是UED，听说过吗？

Tom：哦，是不是国际站的网站布局都是你们来改的？

Heidi：应该是的。

Tom：我是负责客户培训的，我想问一个问题，你们为什么要经常改我们的网站啊，我给客户培训，经常打开网站，就发现布局换了，我的讲义也需要经常更新。

Heidi：这……改动总是有改动理由的嘛（笑）。也许是要提高转化率，也许是为了新的产品上线做拓展，也许是用户的需求更明确了……其实首页平均两年也需要大翻新一下的，老是同一张面孔，多枯燥啊。不过我们每次大的改动，都会通知到可能会受影响的部门，比如销售、客服等，都有相关的邮件抄送给接口人。我回去确认一下你们部门是不是也在抄送名单里。

Tom：嗯，谢谢了。对了，做你们工作的，专业背景都是什么啊？都是学计算机的吗？

Heidi：不是。其实我们部门分四种角色，每种角色的专业背景是不一样的。

Tom：都是什么？

Heidi：用户研究员专业背景可能是心理学、人类学、社科等学科，但是也有其他专业背景；交互设计师专业背景可能是工业设计或其他设计专业，比如平面设计——但是也有完全和设计不沾边的专业，比如心理学也有；前端工程师专业背景可能是计算机，也可能是其他专业，比如我们团队曾经有从演艺圈出来的杰出前端；还有一个角色是视觉设计师，专业背景有可能是平面设计，也有可能是动画设计。啥专业都有可能，不能笼统去讲……

Tom：你刚才说有可能是学工业设计的，我觉得很奇怪，工业设计不都是一些现实的产品吗，比如汽车、洗衣机等……

Heidi：毕竟国内专门的交互设计专业很少，专业的也不多，还是会借鉴很多来自成熟的工业设计领域的一些理论。工业设计可能本身发展比较成熟，而且研究的方向也是人机交互类的，也需要研究用户的特点、行为和习惯，进而设计或者改进产品设计，让它简单易用。所以，和交互设计是有很多相同的部分。我个人感觉交互设计是从工业设计领域衍生出来的，而且并不局限于网站和软件界面，可能在不同的领域有不同的叫法吧。

Tom：那其实，网站、软件和洗衣机设计是同一个道理，只不过是虚拟的和实体的……

Heidi：也许可以这么理解。

Tom：做一个网站很难吗？你们为何需要这么多人？是不是就给程序员一个图，写一些代码就可以了？

Heidi：单纯做一个网站是挺简单的。做好，不容易。做个房地产宣传的网站，容易；做个小企业的形象网站，也容易。做好阿里巴巴、支付宝和淘宝网站，不容易。网站这个概念也不能笼统去讲。越是要用户去使用的网站，设计难度就越高。而且，也并非一个UED的团队在“做”网站，我们几百个人的工作都聚焦在这个网站上。有做产品规划的，有做运营的，有做客户支持的，也有做海外推广的，有做数据分析的——只是大部分人的工作，需要最终流转到我们手里得以分析、实施。（注：做网站看起来简单是因为对于其他人来讲，只能看到20%的工作，但是这20%是靠那些看不到的80%决定的，参见图1。）

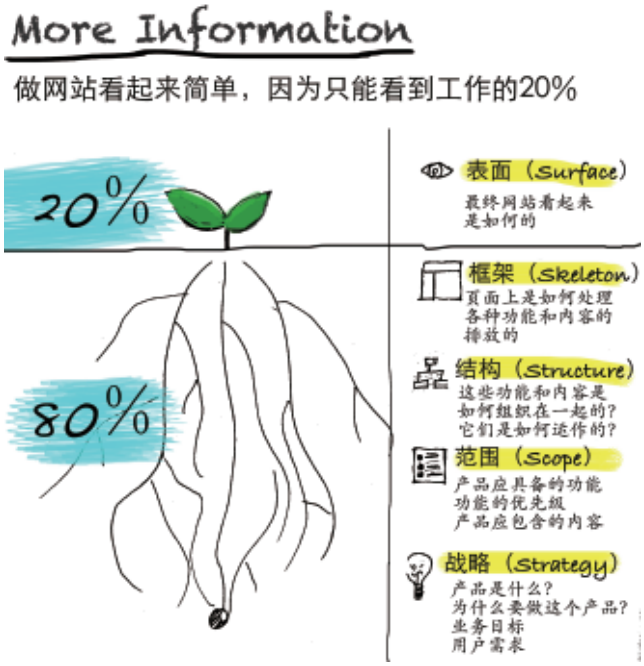


图1　网站用户体验层次：用户看到的是简单的20%

Tom：那你们UED平时都是怎么工作的？

Heidi：做新的产品和优化老的产品的流程其实不太一样。如果是新的产品规划，很早之前就开展工作了，交互设计师、用户研究员配合产品经理（此阶段，大部分是以产品经理主导）做一些目标人群的调研分析，看我们的产品针对的目标人群都是谁、都有什么需求和特点。这样才能够帮我们找到产品的真正定位，确定核心功能以及其优先级等。

Tom：有点像目标市场划分？

Heidi：确实是有营销的分支在其中。其实卖产品要想事半功倍，一定要先有需求才有产品。而不是先生产出产品，再去市场上找能够卖给谁。

Tom：那你们什么时候开始设计呢？

Heidi：刚才说我们前期要投入一些人力去做分析和调研，接下来，在产品经理去细化需要什么功能时，我们就可以开始设计工作了。但是不像你刚才说的，直接去出一张图，而是先由交互设计师做一些线框图。

Tom：线框图，就是草图对吗？

Heidi：你可以理解成草图。就好像大楼要施工之前，也需要有一个蓝图一样。先得把蓝图确认了才能开工。

Tom：这个我理解。做草图是和别人去讨论是吧？视觉设计师设计不也是要出很多稿草图吗？

Heidi：我们设计的草图和视觉设计师做的不一样。我们是在他开工之前更早的一环。但是你说得没错，我们确实可以不出这个草图，而直接让视觉设计师去做。但是这个视觉设计师会非常痛苦，因为他需要花很多精力在草图上，然后他接受很多意见和建议，而且他去优化的方向无从把握，结果导致迟迟确认不了。

Tom：为什么会这样？

Heidi：视觉嘛，每个人都有自己的感受，视觉的评价标准是大部分建立在主观评价上的。你觉得紫色好看，老板觉得红色大方，视觉设计师需要找到一个契合点去让所有决策方达成共识。

Tom：本来就应该这样，即使有你们，他们的痛苦也不会减少多少吧？

Heidi：的确。但是我们的存在会帮助视觉设计师减少大部分痛苦（当然我们存在的价值不限于此），因为我们会缩小他们确认的内容范围。如果没有交互设计的存在，就会把所有的可能性都抛给视觉设计师。在视觉确认时，会发现布局a、布局b和布局c存在各种偏好，而颜色a、颜色b和颜色c也存在不同偏好，版块a的位置，存在不同偏好……结果布局、颜色、结构的排列组合会非常多。（注：如图2所示，视觉设计师的大部分痛苦来源于对各种意见无法控制，从而有效达成决策。）

图2　痛苦的视觉设计师

Tom：嗯，我理解了，会不会有人说：我觉得这个版本的结构很清晰，但是颜色很不好看，你看看能不能把另一个版本的颜色换过来看看？

Heidi：经常！但是这样会诞生出更多的排列组合。

Tom：嗯，想法有时在设计的时候也会收不住，也会诞生出一些自己的新想法。

Heidi：是的。所以我们通常会在交互设计阶段先从理性上确认一些问题，比如布局、信息结构、内容的逻辑性等。这些内容可以不那么感性去评价，更多是靠一些说服性的东西去说服大家接受。

Tom：那你们怎么说服别人接受？

Heidi：如果有数据就用数据；如果有用户的声音就用用户的声音。如果已经了解到用户的特点、行为，都可以拿来应用到设计方案的说服框架里。另外，大家都会尊重专业的观点。知识就是力量，专业让人信任，把自己武装得更加专业一些，会有比较大的帮助。

Tom：就是一样要说服别人？

Heidi：其实有时也不能叫说服，我们自己最起码得相信那就是最好的解决方案，如果自己不确信，就让更多的人去帮你看看是不是有更好的解决方案，如果时间和资源允许，就引入真正的用户去帮我们看看。之后别人都会相信这是目前阶段最好的解决方案。但是，即使交互不像视觉那么感性主观，但是每个人依然都有自己的想法，我们就需要说服他们相信这是最好的解决方案。

Tom：所以还是要找一些支持。

Heidi：数据、用户声音，都很有帮助。实在没有资源，就需要我们去描述使用场景，让其他人觉得自己就是现实的用户，进入到我们描述的使用场景里去，我们把这个说服的方式叫做讲故事。（注：如图3所示，讲故事已经成为一种业界的术语——Storytelling。）

Tom：“讲故事”很有意思！那是不是你们就提供一张页面的草图给视觉设计师就可以了？

Heidi：不但是单个页面的草图结构和布局，最主要的是设计任务流。

Tom：任务流？

Heidi：比如用户的目标是上传产品，为了达到这个目标，他需要经过几个步骤、完成哪些任务等。我们要做到尽可能让用户在最短的时间内通过最少的步骤达到目标。其实我们的目标不只是这个，更高级的是，用户在这个过程中感觉到简单、容易、放松。最有效率有时需要和用户的感受平衡，也需要和我们投入的成本和用户投入的成本相平衡。

Tom：看起来又比较复杂。

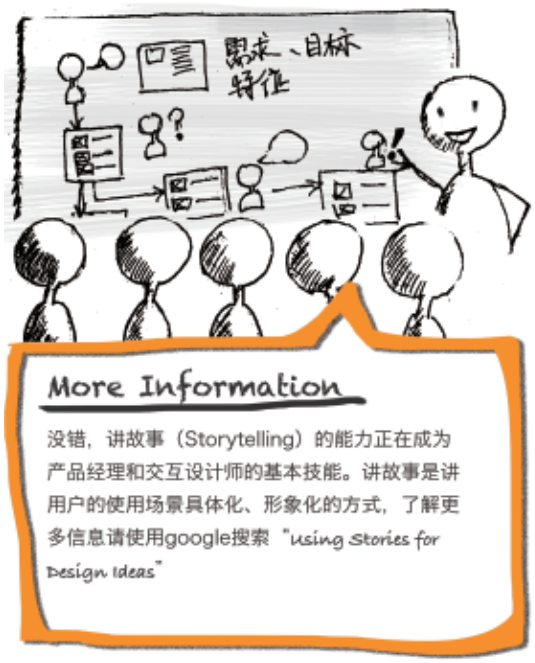


图3 讲故事是将用户使用场景具体化形象化的方式，成为交互设计师说服框架里重要的一种技能

Heidi：其实很简单，打个比方，你在一栋停电的15层高楼里，你需要下楼。最快的方式就是吊个绳滑下来，最舒适的方式是叫个直升机来接你。但是前者虽然效率高但是不安全，你会担心，不敢尝试。后者虽然舒适但是成本太高。

Tom：所以，从楼梯上慢慢走下来，也许才是最好的。

Heidi：嗯，从楼梯上慢慢走下来，对比其他两者，才是最合适的解决方案。我们平时选取设计方案也会有多个维度。（注：如图4所示。）

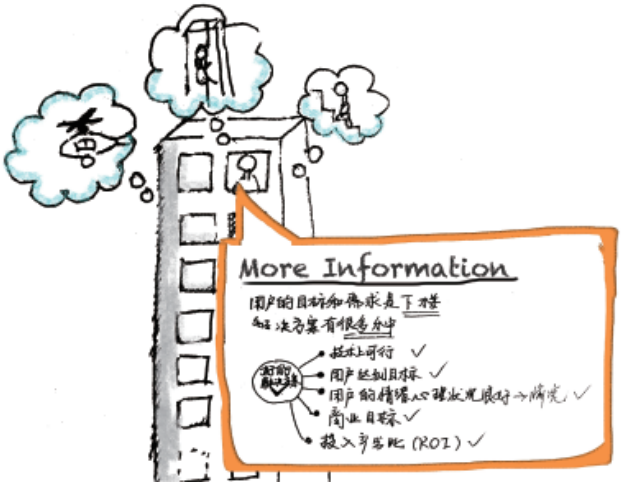


图4 好的设计方案应该能够满足多个维度的评审标准

Tom：我了解了。你刚才说收集用户需求，都是通过什么渠道？其实我们是负责客户培训的，才是每天面对客户的人。

Heidi：客户和用户还不是一个概念。你们培训接触的客户是其中一类用户，也就是国内的供应商，对于他们的需求收集，我们有时也会“上山下乡”，直接派设计师去他们的工作地点去观察他们如何使用我们的网站。当然，我们也会通过内部渠道，比如来自销售、客服部门的反馈。另外，我们也可以很方便邀请他们来我们公司，中国站有专门的实验室，可以让他们在实验室里做一些可用性测试。但是另外一部分用户是海外的买家和供应商。他们离我们很远，他们不容易过来，我们也不能过去。这些用户是我们网站的买家，服务好买家，才能够让卖家受益，所以他们的需求非常重要。

Tom：那你们怎么办呢？

Heidi：只能尽可能挖掘渠道。每年的广交会，国外的买家会来不少，我们也会去广交会参加用户调研。平时有专门的团队，BI部门（网站参谋部）会定期做一些海外买家的问卷调研，并逐渐沉淀出一些买家分层研究，我们可以通过他们知道很多买家的需求和问题。另外，我们自己也偶尔会做一些远程的在线用户调研和测试。

Tom：但是我们网站不会经常推出新产品，你们平时闲着吗？

Heidi：（汗……）我们也想啊，但事实上是我们每天都很忙，而且还在不断招新人。大的战略级的产品不会经常推出，但是小的产品会经常出现，一些老产品的优化不断地在进行——打个比方，刚发布时，注册表单的转化率是50%，之后的优化目标就是将这个转化率提升到70%，目标在不断提高，就需要有后续的行动。

Tom：不能一开始就做得很好，以后就不会经常需要优化了吗？

Heidi：也许快速迭代就是互联网产品的明显特征吧。刚开始大家就是想把产品做到线上去，这时也许本来解决最核心的功能和主要的流程没有问题。我们也无法一开始就做到尽善尽美，一个原因就是资源不允许。另外有可能是，我们确实不清楚到底完美的方案是什么，那些耗费大量资源的功能到底有多少用户需要，所以我们有时就会先发布一个简单版本，之后随着用户使用的数据和访谈的声音，去迭代优化。另外，对于在线的页面、功能都有数据监测，会分析流程中的流失率，我们去判断可以优化的点是什么，然后去改，再监测。

Tom：但是依然没有百分之百满意的时候。

Heidi：每次优化要保证数据是正向的。不是正向的就返回老的版本，继续思考。但是确实不可能说什么是永远不用再优化的。网站的体验就是一点点被优化升级的啊。

Tom：我明白了，谢谢。我之前理解做网站太简单了，想不到这么复杂。

Heidi：哈哈，我们也常说，我们不纠结，用户就纠结。简单是复杂的结果，复杂是简单的必备过程。

总结

以用户的视角来看，我们也希望他看到的都是最简单最愉悦的一面。作为客户培训师，你不需要知道产品数据在我们后台是存储在什么服务器的。用户搜索产品时，也更加不需要了解一个检索词是经过了多少词解析判断计算，跑了几台服务器，最终把结果呈现给他的。但是作为工作人员，我们得了解需要怎么做，才能够把最简单愉悦的结果呈现出来。（注：如图5所示，用一张图来作为本文的总结。）

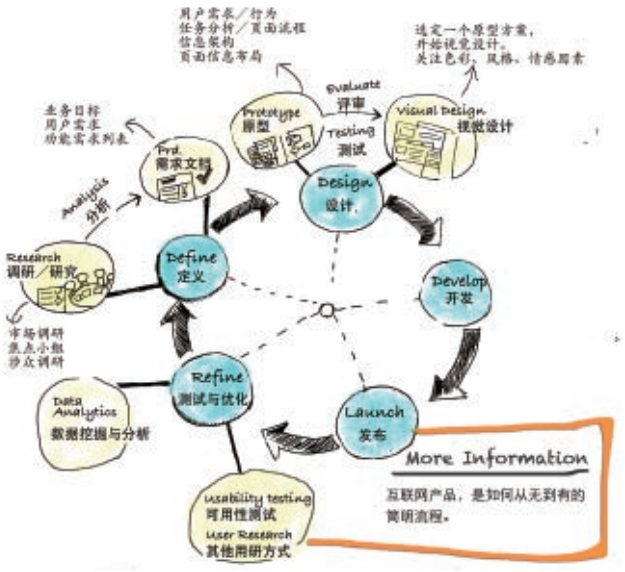


图5　简明网站产品建设流程图

作者简介

谢旭鸿（Heidi），目前供职于阿里巴巴用户体验设计部，从事交互设计师工作。对交互设计、市场营销、产品规划、e-learning均抱有较浓厚兴趣。博客为http://heidixie.blog.sohu.com。

■ 责任编辑：董世晓（dongsx@csdn.net）

一分钟先生
Mr. One Minute

如何把握好分工协作的“粒度”

在工作分配中，分工协作要适当，而不是越细越好。那怎样分工是适当？如何把握好这个粒度呢？

潘正磊

微软Visual Studio
商业软件部总经理

谈到分工协作的粒度，我觉得最重要的是要因人而异，因为员工的能力往往是与职位相匹配的。我给经理的度、经理给主管的度、主管给员工的度完全不一样。微软所遵循的是管理者在与下属确定其工作内容和目标时，要比其现有能力略有一些挑战性，即“踮着脚才够得到”。

微软产品在开发过程中都需要通过反复测试以确保软件质量，以Visual Studio为例，为了确保在不同的操作系统上正常运行各种语言版本，测试往往需要通过一系列自动化测试程序，在几百台机器上历经2~3周才能完成一次。在开发Visual Studio 2010时，我对我的测试经理提出的目标之一是将测试周期从原先的2周减少为4天，但不能增加机器数量，并且要保证原先的测试代码覆盖率。然后，测试经理与他的测试主管们讨论分析，可以从哪几个方面实现这个目标，这就涉及很多协调工作。接着主管们根据下属的现有能力进行工作分配，由工程师们分工研究并完成各项具体工作，例如有的负责虚拟化技术、有的分析如何避免重复运行、有的负责优化测试代码等。最终，分工协作的结果整合在一起达到预期的总体目标。

这个例子说明恰当的分工协作需要做好三个方面的工作：第一，上下级共同确立清晰的任务目标；第二，鼓励员工自发提出可行的解决方案或工作计划；第三，在执行过程中，上下级之间及时沟通、调整、工作进度及处理所存在的问题。员工可能会认为他需要3天完成某项工作，而主管可能觉得是1天或者5天，这就需要员工与主管进行沟通，可能会有三个结果：工作确有特殊因素，需要3天；员工将简单问题复杂化，在主管的帮助下，少走很多弯路，1天足矣；员工将复杂问题简单化，其实需要5天。

作为主管或经理，首先要做到用人不疑；在设定工作时，把握好大局和下属正确的工作方向，并鼓励下属发挥主观能动性，提出可行的计划；在执行过程中，结合考核及时与员工交流、反馈，在他们遇到困难时，及时提供帮助，扫除障碍。

崔海

NHN中国技术开发中心总监

对于研发管理者来说，一方面，我们希望每个人有机会可以做不同的模块或者是不同的产品、不同的技术，这样他可以去学一些新的方法和技术，从而有机会做不同的工作；另一方面，我们又希望员工在做研发的时候，要有一定的深度，需要对他所在的专业领域有较多的积累。从某种角度来讲，这两个方面是互相矛盾的，所以说我们需要从中寻求到一个比较合理的平衡。

首先，在一个既定的时间周期内，我会安排某一个人专注于开发，不会在这个时间周期内进行频繁的人力变更或者某一个领域的负责人的变更，这样就能够保证他对自己所开发的产品或模块有一定程度的熟悉度和关注度。

其次，在具体的项目分工过程中，我认为整个项目的工作进度是由这个项目的执行计划来确定的，一般的开发流程是这样的：首先，在项目启动之前，要制订好项目的策划和企划，策划和企划是不一样的内容，策划侧重于产品是什么样的、它具体做哪些事情等，企划则侧重于产品计划、产品何时上线、公司需要投入多少资源、具体的规模等；在策划、企划之后，需要对产品的需求和技术设计文档都有一定程度的明晰，这样任务的分配就不是一个无序的分配，而是根据业务的逻辑进行了有机的划分，使得每个人负责的都是一个相互之间有联系的模块。

最后，一般一个项目会安排一个主要的负责人，负责项目中的一些比较重要的决定，包括项目内部的分工协作等。分工协作时需要把握的一个基本原则就是让员工做他擅长的事情，而且要分配给他一些富有一定挑战性的工作，经过这样的良性循环，就会使他处于一个迭代的、逐步提高的成长过程。

关注IT人自我管理、职业发展，广邀经验人士排疑解惑。

花一分钟阅读，还你一个明白。

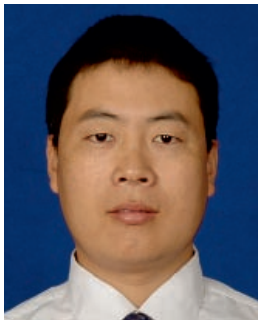
冯国馨

联合永道副总裁兼CTO

分工协作是否越细越好？不一定。这个问题分组织级和项目级两个层面。组织级涉及组织规模、组织管理模式，项目级主要是面向项目成员个体对象。下面结合组织级来谈。

组织规模分为两种情况：成熟度高、规模大的单位和成熟度相对低、规模中小的单位。前者，组织结构庞大，资源相对充足，分工协作细致，规章流程相对成熟，协作的要求是任务尽量明确可依。以IT企业为例，这类企业为了保证目标和任务的有效达成，通常会有明确的任务书、规格说明书、系统模型架构设计、模块设计、用例方案等细化的文字工作产品作为分工协作的依据。此时分工协作的度，就是要尽量界定清楚目标及任务的范围、边界和接口定义，避免产生责任不清的问题。而后者，往往资源不是很充分，没有很规范的规章制度可循，对人的素质和能动性要求相对比较高。同时，中小型公司面对的客户通常也相对不够成熟，客户变更和要求会相对多，任务也不易明确。此时，项目的分工协作就相对粗一些，通常给出目标和任务说明而不会有太多细化的文字，主要通过项目经理对项目成员能力的把控，相对粗粒度的工作安排和密切的管控来保证任务的完成。这也是中小企业在项目管控上容易出问题的原因，但它却对整个项目组成员都十分锻炼，能者就有机会被提拔而且提升很快。

组织管理模式也非常影响分工协作的模式，不同的项目组织结构使得项目沟通形式有很多不同。项目组织结构分职能型（例如软件部、测试部、质量保证部等）、项目型和矩阵型。作为组织级的项目总监需要界定好项目在不同部门流转的规则、输入条件，余下的需要各部门经理按部门能力确定分工粒度：职能型项目经理的主要职责是将组织级的要求固化执行保障好即可，通常分工粒度没有权利；项目型项目经理具备生杀大权，操作模式参照组织规模的方式；矩阵型项目组有部门经理和项目经理两个头儿，组织级分工协作挑战最关键的工作就是授权和流程定义，此时项目经理基本也能参照项目型项目经理进行管控。



史永锋

奥博杰天项目技术经理

团队合作的价值在于通过团队成员之间的默契合作，实现1+1大于或等于2的工作目标。实践中，绝大多数的工作任务需要根据任务本身的特点和团队成员的能力进行粒度划分。

粗粒度的任务划分会增加项目监控的难度并影响任务估算的精度。过粗的任务划分导致每个任务所需的工作时间较长，而整个项目的进度反映在每个任务的完成情况上，因此这种划分大大降低了项目进度监控过程的实时性。此外，过粗的任务划分由于对任务实现的过程考虑不够详细，在后期估算过程中容易出现估算不精确的情况，从而增加项目的时间风险。

任务划分的粒度如果太细，则任务之间的依赖性、耦合性和优先级管理变得相当复杂，团队成员花费在任务管理上的工作量相对于任务本身的工作量更为突出，因而使得任务完成的效率大大降低。

合理的任务划分粒度可以大大提高任务完成的效率。任务划分的粒度，一般以划分后的任务能在1~20个小时内完成为佳。低于1个小时的任务需要更多的时间用于任务协调；超过20个小时的任务则会增加任务按时完成的不确定性，进而影响整个项目的研发周期。

划分后的任务分配给团队成员时一般使用两种分配策略：团队领导主导的任务分配策略和Scrum抢任务式的任务分配策略。前者由团队领导结合任务在时间、优先级、业务和技术等方面的要求将任务分配给最适合的团队成员，从而使任务安排满足项目的时间需求。这种方式效率很高，缺点是如果某个团队成员对自己分配的任务不感兴趣，则会影响其工作积极性和长期的稳定性。后者则通过抢任务的方式使各团队成员分到自己感兴趣的任务以便充分调动工作积极性。这种方式单个任务的完成时间要多于第一种策略所需的时间，但由于团队成员总在做自己感兴趣的任务，将有助于保持团队的长期稳定性。在任务估算时间少于任务可用时间时，可以适当地采用抢任务的方式进行任务的分配。

主持人

冯大辉，现任丁香园（http://www.dxy.cn）网站CTO。曾历任支付宝架构师、数据库团队负责人等职。

架构师接龙

侯震宇 vs. 潘晓良

提问嘉宾 侯震宇

侯震宇，百度基础平台部架构师。2003年毕业于中科院，曾参与百度贴吧、知道、空间、搜藏等产品线的设计和开发。现在负责百度基础平台的规划及建设，专注于支持超大规模数据量和并发访问量的分布式平台建设。

回答嘉宾 潘晓良

潘晓良，百姓网技术总监、联合创始人。曾任职于国信朗讯、毕博全球开发中心和 eBay，在百姓网成立之初加入团队，组建了百姓网的技术团队，现在负责百姓网的开发和运维，以及技术人员的培养和发展。

侯震宇：在编程模型上，您认为如何能够更好地利用多核以及多机环境？

潘晓良：我是把它分成两个问题来看的：一个是多核问题，另一个是多机问题。

对于多核问题，我是一个实用主义者，真的不想在上面有所投入。我把多核环境认为是操作系统应该完成的事情，相信微软和Linux的那些内核朋友们都在为此而努力，因为他们是专家。但看着几个核的资源分布严重不均，一颗累死，十一颗闲死，资源实在太浪费了，怎么办？让系统工程师来搞定吧，就是多跑几个进程，最简单的例子就是一个机器上起多个Memcached、多个Search、多个MySQL，既然操作系统无法把一个进程很好地分到多个核上去，那么我们就用人为的方式来做一些，这虽然不能彻底解决多核效率的问题，但是至少缓解了一点。而且据我所知，很多公司都是这么做的。

对于多机环境，按我的理解，问题只有一个，就是数据访问必须是共享的，对于一个Pool中的Web服务器，它不能使用Session，也不能把数据放到本地文件中，因为不知道下一次访问会不会命中自己，包括本地缓存，随着Pool中服务器数量的增加，命中率随之下降。虽然以上的问题可以通过负载均衡的固定Session方式或者URL映射的方式来做，但是它可能造成服务器的压力分配不均，后面带来的问题可能更多。所以我们的方式就是禁止直接使用Session等本地资源，使用上面一层的虚拟对象，由虚拟对象来控制数据访问，而使用共享的Cache来实现Session的基本功能。

侯震宇：在纯代码层面上，您的团队是否有规范以确保大范围的代码组织结构是统一的。比如有项目的lib、include目录和基础库等。你们是否实现了一个统一的编码编译环境？有什么困难？

潘晓良：对于百姓网来说，因为我们只有一个产品，基本上还没有多项目的问题，但是我想这个问题的关键还是在编码规范上，在于代码的质量和不断增长的代码数量之间的矛盾的问题。关于这个我们做了以下几个事情。

由Developer们讨论出来的编码规范。规范的事情是这么写也对，那么写也对，但就是要争取大多数一致，所以我们的规范是Developer们一起讨论出来的。

标明代码责任人。以前我们的代码是按照模块划分的，但是人会换模块，后来就会多出来很多无主代码，这些无人维护的代码，不仅容易出现Bug，而且增加了阅读成本，所以必须清理掉。

每日上线的时候把每个人的代码数量统计发出来。我们认为代码的少就是多，每个人有责任控制自己代码的数量，是多是少大家都要看见。

每周固定时间Pair Programming。代码质量的重要性大家都知道，但是给予的时间却不多，我们有固定时间段用于Pair，一来可以加强开发人员沟通，二来可以提高代码质量。

给实习生一个平台。实习生很有Passion，而且有很多自己的想法和独特的视角，其实他们在学校里面学了很多东西，只是缺乏应用，我们的实习生可以访问所有代码、修改任何一行代码，只要稍加指导，他可以做任何他愿意做的功能。

统一的编码编译环境非常重要，必须实现。互联网软件开发依赖的环境很多，数据库、搜索、缓存、定时脚本、消息队列、Web服务器等多种服务，要是自己搭建肯定是事倍功半，一人发几台机器都不行，不仅资源浪费，而且环境不同，也会带来很多意想不到的Bug。我们在实现了统一环境之后，每个人都只需要用一个笔记本开发就可以了，既节省了资源，也极大地方便了Pair Programming，方便了工具的统一和思想的交流，也方便实习生的学习和融入。

实施统一编码环境的障碍肯定不是技术，最大的困难还是人，首先需要团队的Leader支持，其次需要有好的方法分步骤实现。开发人员都会认为自己的习惯很好，不愿意改变，所以只能先小范围尝试，让愿意改变的一部分人先满意，培养他们的幸福感，然后他们就会去影响其他人，再来几次Pair以后，就慢慢都被同化了。

侯震宇：在大规模机器集群上，你们如何实现程序发布上线、配置的管理以及程序和机器的监控和健康检查？

潘晓良：百姓网是每日发布产品的，只要是工作日就会有版本发布，但我们不是从第一天就是这样的，我们是从每周发布一次，到每周发布两次，到现在每天发布一次的，整体来说，每天发布让我们的迭代周期缩短，加快了产品更新的速度，但是也会带来很大的问题，比如Bug增加等问题。每日上线其实对我们来说是一个很大的挑战，有很多工作需要做。

自动化打包。百姓网是使用PHP开发的，可能有人说不需要Build，但是打包过程可以做很多事情，压缩、去注释、代码统计等很多工作都是在这个时候完成的。

自动化测试。这个过程非常关键，必须通过所有的Unit Test和重要流程的测试之后才能发布，如果出现重要流程失败，则需要找相关开发人员找到问题，然后重新打包，这是基于打包全自动、够快的前提的。

部署上线。那个时候就是一个按钮的工作了，当然这个时候最重要的就是确认上线中是否有问题，即使再详细的测试，也不能保证线上环境能够顺利，所以上线之后我们就要关注关键功能是否有问题，我们有一个Rollback的标准，达到这个标准则必须Rollback，如果达不到这个标准，则直接在线上进行Hot Fix。

关于配置的管理，也存在于版本控制工具中，比如叫config.online.php，在自动化打包的时候，会被命名为config.inc.php，通常情况下，线上环境配置是不需要更改的，而在配置更改的时候，只需要修改它就可以了。

实施统一编码环境的障碍肯定不是技术，最大的困难还是人，首先需要团队的Leader支持，其次需要有好的方法分步骤实现。

关于程序和环境监控的问题，我们基本上采用Cacti、Nagios这样的主动和被动监控工具，但是它们都是在系统级别上，除非是程序导致系统压力有巨大变化的状况，程序中的问题就不能被发现，所以我们还有程序级别上的监控，最主要的是监控错误日志，主要是PHP错误，如果有错误，那么开发人员很快就能收到邮件，可以及时修复。

侯震宇：在目前国内的网络环境下，一个分布式系统是否需要考虑多机房问题？如果需要，主要考虑哪些问题？

潘晓良：关于多机房，我觉得完全取决于业务需求，即访问速度是否影响了业务，如果影响了，那么南北机房应该是最有效的方式，但是我觉

新闻 NEWS

SAP推出全新业务分析应用软件系列

10月13日，SAP公司宣布正式在中国市场推出全新的SAP Business-Objects业务分析应用软件系列。该业务分析应用软件系列旨在通过创新的业务分析功能帮助企业用户在任意地点、以任意方式获取关键的行业性信息，包括一整套能够有效解决特定行业问题和挑战的解决方案，并且适用于不同业务环境。

诺基亚宣布Qt为Symbian和MeeGo唯一开发环境

诺基亚官方网站宣布，将采用Qt作为未来Symbian和MeeGo平台的唯一开发架构。诺基亚将向开发者提供更加统一和简明的开发工具，以鼓励开发者社区为Symbian和MeeGo平台开发应用。诺基亚计划将Qt作为唯一的应用开发架构，以保证应用同时与未来版本Symbian和MeeGo系统兼容。另外，诺基亚宣布将在Symbian和MeeGo平台支持利用HTML 5发展网络内容和开发应用。为展示诺基亚对新技术的支持，诺基亚将使用Qt开发未来应用。

Mozilla推出应用程序商店平台

10月20日，Mozilla公司推出了一个“开放式网络应用程序生态系统的原型”。Mozilla称这个生态系统为“Open Web Apps”（开放式网络应用程序商店），并已经制定了关键的技术文件和设计原则。Mozilla不希望网络应用程序被局限在某个应用程序商店、浏览器或平台上。根据Mozilla的倡议，开发人员可以直接将网络应用程序销售给用户，也可以通过任何一家网络应用程序商店进行销售，而且这些应用程序必须可以在所有兼容它的操作系统，以及移动和桌面设备中运行。

得这里面的成本还是很高的，一个机房变成两个机房，就如同是一台Web变成两台Web，对于系统架构的影响比较大。百姓网是经历的这样的过程的，我们考虑多机房是完全基于业务发展的需要，不过多机房也没有那么可怕，就是要多付出一些成本。

硬件维护成本。通常都是异地，有人驻扎还稍好，无人驻扎，正常的机器维护都会成问题，我们的解决方案就是使用刀片服务器，虽然硬件成本上升了，但是少了人员维护成本。

可用性风险。我不知道有多少公司会让两个机房有相同的承载能力，能够以一个机房之力承载所有请求，因为那意味着大量冗余，所以肯定是各自承担一部分的压力，但是宕机的风险却加倍了，而且还引入了另外一个风险，就是机房之间通信的风险，网络的延时可能会对业务造成很大影响，如何消除延时或减小网络延时对于业务的影响就是最关键的一点。

开发、部署、调试的成本。代码部分会产生一些修改以支持多机房，部署的时候两边都要走，即使是自动化发布，也需要写脚本，特别是调试的时候，更加需要知道是哪个机房的问题，我们曾经遇到过只有一个机房出Bug的情况，调试的时候等于说是增加了一种因素。

多机房看似只是多了一个机房，但是对于运维来说，很多工作就变得复杂很多了。

侯震宇：对于机器选型问题，是为不同的大型专有系统（如分布式存储）专门配置的机器，还是统一采购同一类型的机器将不同的系统共享一个物理集群？这两种方法有什么优缺点？如何选择？

潘晓良：不同类型的机器的好处是度身定制，物尽其用，比如存储的机器的硬盘大，缓存的机器内存大，Web机器可能就用最小的硬盘等，虽然看起来很美，但是不灵活、不通用，多出来的大硬盘机器，其他地方用不到，反过来小内存的机器又不能给缓存的机器用，特别是在做系统调整的时候，很多机器因为不合适而不能用了，反而容易造成资源浪费。

相同型号就是完全反过来，它有明显的好处——通用，放在哪里都能用，但是问题就是可能对于某些特殊应用不是很适用，做存储的话硬盘小了点，而做Web服务器可能硬盘又大了点。

对于这个问题，百姓网通常还是会选择用一种型号，然后再对不同类型的服务器去升级硬盘或者内存，而CPU等核心的内容基本上一致，这样就既能保留通用性，又有一些灵活性。不过我们还是会在一些特殊应用上采购一些特殊机器设备，比如数据仓库，它们对于容量和计算的需求就会超过正常的机器的范围，可能就有一些特殊的要求，我觉得这里面还是一个平衡的艺术。

不过我们觉得这种灵活性还是相对的，因为机器的升级速度太快了，通常1~2年就有一种新升级型号，之前的内存硬盘都有可能很难买，或者价格很高，这时候就宁可选择新型号，而把旧型号放到一些非关键应用上。

以上是我们在使用过程中的一些感想体会，不能说放到哪里都准确，欢迎技术朋友们发邮件过来指正，我的邮箱是panxiaoliang@baixing.com。另外，我们也非常欢迎喜欢互联网的技术人员或者同学来百姓网工作实习，我们的职位发布在http://jobs.baixing.com/。

选择编程语言就像选择酒吧

——Joshua Bloch访谈

■ 文 / Peter Seibel　译 / 郝培强

本文是Commmon Lisp专家Peter Seibel对Google公司首席Java架构师Joshua Bloch的访谈，谈到程序员应该看什么书、如何能快速熟悉一种新语言以及为什么说选择编程语言就像选择酒吧。

Seibel：你是怎么开始编程的？

Bloch：我想这是受益于我的家庭影响。我父亲是Brookhaven国家实验室的化学家。当我上小学四年级的时候，他参加了一个程序设计培训班。当然在那个时候，电脑都是放在玻璃窗背后的大型机，你只能把写好的程序卡片交给操作员。虽然没法儿亲自动手，但我还是被电子计算机可以帮助你做事儿这一点震撼了。所以，我在父亲上课的那段时间，跟他学了一点儿Fortran。

Seibel：那大概是哪一年？

Bloch：我想是1971年。直到很多年以后我才真的对程序产生了强烈的兴趣。让我产生兴趣的当然是分时系统。长岛有一台DEC system-10电脑，供Suffolk县内所有的学校使用。Nassau县也有一台。很神奇的是，很多著名人物的事业都是从这两台DEC system-10电脑开始的。

你的程序一旦有交互，就会有Bug。大概是从1973~1976年，那时候我跟其他人一样，在写BASIC程序。我就是从那时开始正式写程序的。你知道吗，我还保存着当年写的程序，是印在电信打印纸上的。如今回头再看这些程序的时候我发现，我代码风格中的某些部分从那个时候起就一直没变过。

Seibel：你还记得你写的第一个有趣的程序是什么吗？

Bloch：噢，我记得那是1977年7月4日，我为经典的二十问游戏写了一个程序，叫“猜动物”。这个程序包含一个二叉树，是非题位于它的内部节点，动物位于它的叶节点上。如果用户所提的动物是叶节点上没有的，它会向用户提出是非题，通过区别新动物和它猜出的错误动物之间的差异来了解新动物。二叉树保存在硬盘上，这样程序可以越来越“聪明”。

我当时想，“天啊，真酷，程序真的能学习。”这是我一生难忘的瞬间。我还记得另一件事。当时我在高中，应该是10年级吧。是关于DEC system-10的。当时不允许我们编写现在叫做即时消息软件的东西，因为它们对系统资源的消耗实在太大了。

Seibel：如果时光能够倒流，可以一切从头来过，有什么东西是你真的希望改变的？Basic对你来说太简单了？其他还有什么？

Bloch：我没什么遗憾的，实际上Basic很有趣。我觉得Dijkstra对Basic的看法是完全错误的。原谅我这么评价已故的人，愿他在天堂安息。我知道很多非常好的程序员，他们是从BASIC编程开始的，因为那是他们能找到的唯一的语言。

然而我觉得使用多种语言是件好事。上大学的时候，我用很多语言编程。每门课都可以用一门语言。在数学课或者理工科课上，应该用Fortran。那时候编程课学的都是Pascal、SAIL、Simula或者类似的东西。在人工智能课上，用LISP。

不过也许我应该学更多的语言。有意思的是，一开始我对面向对象并不感冒，直到那个二十问游戏开发快结束了，我才真正对面向对象有了感觉。严格来说，Java才是我真正使用的第一种面向对象语言，某种程度上是因为我不太想用C++。

Seibel：那是什么时候？

Bloch：那是我1996年加入Sun公司时。我觉得要是我能更早学习这些概念就好了。我不认为这些概念都是好的。面向对象很有意思，它有两层含义。第一，它意味着模块化。模块化是非常好的。但是我不认为这是创造面向对象的人们的专利。你可以去看以前的文献，例如，Parnas关于信

息隐藏的论述，就会发现这种概念可以看作面向对象编程中类概念的一种抽象的原型。第二，它意味着继承，我认为继承有利也有弊，这跟如今很多人的使用感受一致。

另外，我应该进入更多的领域，计算机科学领域内外都应涉猎。你学的东西越多，开始得越早，对你越好。我一直没有真正做过的就是GUI编程，在某种程度上说我应该强迫自己做做看。但是由于种种原因，如这些年来开发库代码，构建他人可以使用的代码块，这些事情已经占据了我的大部分时间。这样算来，我做数据结构和算法等方面的工作已经有几十年了。

Seibel：有什么书是所有程序员都应该看看的？

Bloch：《设计模式》无疑是一本，虽然我对它的感情有点复杂，但是我还是认为每个人都应该读一下。书中列出了通用的词汇，也提出了很多好的创意。另一方面，这本书有点儿像方法和语言的大杂烩，内容也有些过时了。但是我认为它绝对值得一读。

另外一本书是《Elements of Style》，它甚至不是一本编程书。为什么要看这本书呢？理由有两条。首先，每个软件工程师工作中很大一部分是写文档。如果你无法写出精确、统一、易读的说明书，那么没人会去用你的产品。所以说可以改善你写作风格的东西都值得借鉴。其次，该书里面的大部分思想都适用于编程。

我的荒岛列表有点古怪。例如，最重要的书是Herry Warren写的《高效程序的奥秘》（Hacker’s Delight）。

Seibel：这是一本位操作（bit-twiddling）书？

Bloch：是的。我爱位操作，这跟我的工作有关联。如果你写库、编译器、底层图形代码或者加密代码，这本书是不可或缺的。Warren把曾经口口相传的东西放在一起，用严谨的数学去进行验证。这本书出版的时候，我被震惊了。

当然还有Knuth的《计算机程序设计艺术》。事实上，我从来没有读完这一套书，至少没有从头到尾看过。但当我研究某个具体算法的时候，我就去看他会怎么说。往往可以得到我想要的东西，这套书太全面了。

但是我没有能力、也没有时间去读完整套书，所以如果我告诉你我读完了，那么我就是在说谎。我觉得还有一本非常好的老书，是Kernighan和Plauger写的The Elements of Programming Style。书里面的例子都是用Fortran IV和PL/I写的，所以有点过时。不过，虽然这本书这么老，但里面的思想却从未过时。

另外一本老书是Frederick Brooks的《人月神话》。这本书都出版40年了，里面的思想仍旧同出版时一样有影响力。阅读它是一种快乐，每个人都应该读一下。这本书的主要信息是“给一个延期的项目加人，会让它延期得更加厉害”，今天这一点仍旧是正确的。里面还有其他很多重要的观点。有些细节虽然过时了，但仍值得一读。

现在每个人都必须要学习并发编程。所以应该看看《Java并发编程实践》这本书。虽然标题中有Java，但是很多内容并不限于任何具体的编程语言。

Joshua Bloch，现任Google公司首席Java架构师。之前他在Sun公司工作，领导并实现了Java 2中的Java Collection Framework，还参与了Java 5发行版中几项语言附加特性的设计。Bloch在卡内基梅隆大学获得博士学位，期间他参与设计了Camelot分布交易处理系统，后来演变为Transarc公司的产品Encina，而他则是Transarc的资深系统设计师。他编写的Effective Java一书获得了2001年Jolt大奖，他还与人合著了《Java解惑》和《Java并发编程实践》。

Seibel：这就是你和Brian Goetz合著的那本书？

Bloch：我的名字是印在封面上的，但是我提到它的原因恰恰是因为这不是我写的书。第一作者是Brian，第二作者是Tim Peierls，制定Java并发标准JSR-166的每一个人也都是这本书的作者。把我的名字印在了封面上仅仅是出于礼貌，我贡献了些材料，但是没有正式参与编写此书。

噢，还有一本：《韦氏学院词典（第11版）》。我去哪里都带着它。这倒不是你实际上要读的东西，但是我说过，写程序的时候，必须能命名好变量。你的文笔必须好。没有好的字典，我就会觉得少了点儿什么。

Seibel：除了命名好变量，尽量少的复制粘贴以外，你经验丰富后，还有哪些写程序的习惯改变了？

Bloch：随着年龄的增长，我逐渐意识到编程不仅仅是让程序运行而已；编程是创造一个易于理解的、可以维护的、高效的作品。一般来说，我发现，干净整洁的代码，往往运行起来更快。这与流行观点正好相反。而且即使它们不快，也可以很容易地让它们变快。正如人们所说的，优化正确的代码比改正优化过的代码容易多了。

我的一些改变是跟具体语言相关的。每种语言都提供了一个工具箱。你要使用正确的工具，在这种语言中正确的工

具在另外一种语言中可能不是最好的。举一个简单的例子：如果你用Java 5，使用枚举来代替整数常量或者布尔常量可以大大简化程序，让它更安全，更可靠。

Seibel：说到这儿，你能否谈谈如何能快速熟悉一种新语言？

Bloch：这跟人类的语言很类似。一种方法是学会很多语言。如果你已经熟悉意大利语和西班牙语，那么学葡萄牙语就不需要花太多时间了。你知道的越多，你能吸收的就越多。

学习一种新语言的时候，要利用以前所学的语言的功底，但是也要保持开放的心态。有些人执著于一种理念："这就是写所有程序必须遵循的方法。"我不说是哪种语言，但是某些语言，出于某种原因，令人执著于这样的理念。当他们开始学习新的语言的时候，他们批评这种语言跟真正的神的语言的所有不同之处。当他们使用新的语言时，他们极力使用真正的神的语言的方法去写。这样，你就会错过这个新语言真正的独特之处。

这就像你本来只有一个榔头，有人给了你一个螺丝刀，你说："唉，这不是一把好榔头，但是我应该可以倒着抓住螺丝刀，用螺丝刀把来砸东西。"你得到了一个很烂的榔头，但事实上它是一把很不错的螺丝刀。所以你应该对所有事物保持开放和积极的心态。当然，最重要的是代码！代码！代码！要多用语言，这样才能学得更快。

Seibel：为什么人们对所选的计算机语言那么虔诚？

Bloch：我不知道。但是选择一种语言时，所考虑的不仅仅是一系列技术上的权衡，而是在选择一个社群。这就像选择一个酒吧。没错，你希望去一个提供美酒的酒吧，但是美酒不是最重要的。主要是那个酒吧里都有什么样的人，他们在谈论些什么。选择计算机语言也是这样的。时间一长，就这门语言也形成了一个社群，社群里不仅仅有人群，还有他们的软件成果，如工具、库等。这就是有些理论上看起来更好的语言无法成功的原因，他们无法在周围构建成功的社群。

Seibel：要是这么理解的话，Java非常有趣，它有两个社群。一个是实现者和系统开发者，也就是在Javasoft、Weblogic或者类似地方工作的人们。另一个是所有用Java、应用服务器、构建好的框架来构建商业应用的人们。这两个酒吧差别非常大。

Bloch：与Java或者其他语言关联的社群很多。如果语言周围没有形成社群，这通常表明，要么这个语言无人问津，要么就还很不成熟。语言繁荣发展，就会自然地表现为出现越来越多各式各样的社群。同时，如果对一门语言的投入总额增长了，那么它的价值也会相应地增长。

这就像梅特卡夫定律（编者注：一种网络技术发展规律，由3COM公司创始人、计算机网络先驱罗伯特·梅特卡提出。）：网络的价值与用户数量的平方成正比。这对于编程语言也适用。所有使用这种语言的用户构成社群，然后突然间出现了Eclipse，出现了FindBugs，出现了Guice。即使Java对你来说不是最好的语言，但是使用它有这么多附加的好处，所以你还是会创建社群，解决如何在Java中进行数学编程，或者你需要的其他类型的编程方法。

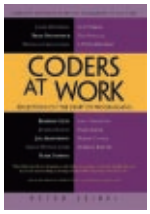
Seibel：你有没有过这样的感觉：你越来越清楚那程序就是搞不定，到处都是这样那样的问题，几乎令人绝望？

Bloch：当然有过。写书也是这样的。每次开始一件事情的时候，我总想逃避。开始是最艰难的，有时候我会鼓励自己："加油Josh！这行你都干了三十多年了，你知道怎么能做到跟别人一样好，不必瞻前顾后，动手吧。"或者对自己说："你瞧，以前的一切你都做的挺好的，这次也错不了。"

Seibel：你刚才提到你的生活体验变宽广了，这可能会影响编程，但是有没有什么东西，是编程以外的体验，但是帮助你成为了一个更好的程序员呢？

Bloch：当然有。我认为你能做好的每件事情都有这个作用。思想是没有学科限制的。我想到了一个例子。我写论文的时候，要对一种分布数据结构，RSM（Replicated Sparse Memory，复制型稀疏内存）做一个分析。而做这个分析的基础思想来自于我所上的化学课。那是一个动态平衡公式：如果系统里有一个动态的平衡关系，就可以写出一个等式，"事物进入一个特定状态的速度，和他们离开这个状态的速度相等。"这样就得到了三个变量的三个等式，求解这三个等式，计算出来的结果和观察到的这种复杂的分布数据结构的行为恰好匹配。这就是我直接从化学里偷来用于计算机科学的思想。

你在生活中看到的很多东西，不管是架构上的，即建筑物构建的方法，还是在语言上的，即人们进行沟通的方法，很多思想都是可以借用的。当然包括数学。数学和编程相当类似。所以要睁大双眼，积极地吸收重组各种思想，这样做绝对错不了。



本文节选自《Coders At Work》一书。该书是当今15位大师级计算机程序员的访谈录，重点介绍了他们的编程感悟。中文版将由人民邮电出版社北京图灵文化发展有限公司出版，特此感谢图灵公司授权。

云计算的资源使用分析

■ 文 / 方国伟

本文首先把传统IT资源使用方式与云计算资源使用方式作了对比，分析了云计算的资源使用模型的特点和几种适合云计算模型的应用场景，对比了不同计算模式下的差异，并从资源利用率的角度展望了云计算未来的发展趋势。

云计算无疑将改变整个IT生态系统，IT服务的提供模式以及资源的利用方式都将产生巨大的改变。云计算的发展有两个最为重要的推动力，同时也是它最为根本的两大价值，一个是帮助用户降低IT服务成本，另一个是提升用户使用IT服务的体验。云计算之所以受到广泛关注，与全球经济的发展状况有很大关系。当经济状况变得具有挑战性时，人们对成本敏感度就会上升，企业IT方面的投入就变得更加谨慎。因此，当云计算可以帮助企业降低成本的时候，大家对云计算所爆发出来的热情也就不难理解了。当然，云计算还有许多不同方面的价值，下面我们从资源利用的角度分析一下云计算的资源使用模型及其适合场景。

云计算的资源使用模型

要理解云计算的概念以及它能带来的好处，还需要理解云计算的资源使用模型。传统的IT使用模型是根据业务发展情况预计一个应用系统的负载增长率，然后制订一个固定时间比如3~5年的投资计划。在这种方式下，用户需要一个大的前期投资，以满足一定时间段的用户需求。这种前期投资对于小企业和初创公司来说就是一个很大的进入障碍。就算在理想的状况下，也就是说应用系统的负载量与预期负载一样，这种构建方式也会造成IT能力的浪费。而实际情况是应用系统的负载很难做到精确，这样就更加剧了IT能力供给和需求之间的矛盾。实际系统的IT能力的过配和短缺变得相当普遍。一些大型用户为了保证关键应用的正常运行，系统过配现象就非常严重。这种传统的IT资源使用模型如图1所示。

云计算的出现使得我们可以用更为灵活的方式来配置系统资源。一方面由于采用新的租用的业务模式，因此用户的进入门槛相对较低，不需要有大量的前期投资就可以开始使用资源。另外，云计算可以根据实际需求来进行资源调配，因此用户不用担心高峰时期资源的短缺，也不用担心系统在压力小的时候资源会限制。这种弹性的资源配置使得用户的投资能够取得最佳回报。云计算的资源使用模型如图2所示。

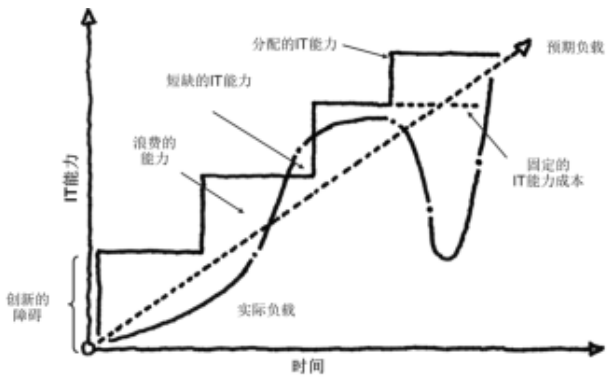


图1　传统IT的资源使用模型

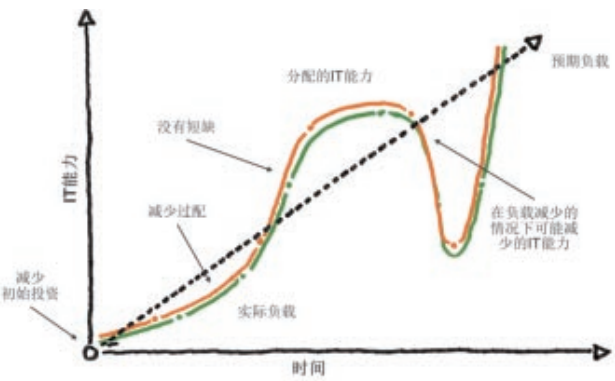


图2　云计算资源使用模型

适合云计算的典型场景

许多人在一开始接触云计算的时候，往往倾向于认为云计算是无所不能的。虽然我们认为云计算是未来IT发展的方向，但是在云计算发展的初始阶段，不是所有的应用都适合云计算的方式。根据云计算服务的特点，下面几个应用场景是一些可以从应用云计算中得到最为显著的效果。

间断性应用场景

间断性应用场景包括临时性的一些应用需求或者一些批

处理工作等。这些应用场景的特点是不需要连续地使用计算服务。比如，企业要进行一场为期一周的市场推广活动需要使用计算服务，但是通过传统方式来处理这类计算需求就显得笨拙，一方面响应比较慢，另一方面还会造成许多资源的浪费。云计算通过按需分配的方式可以比较好的符合这类应用场景。如图3所示。

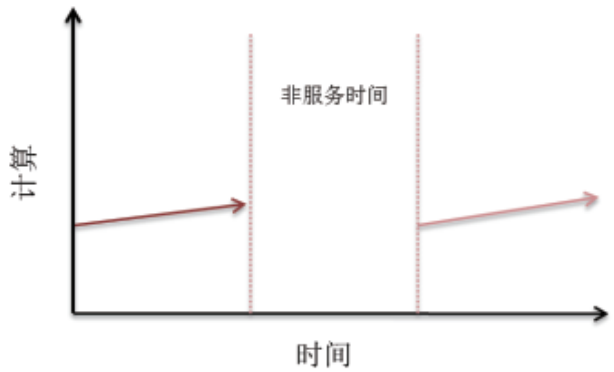


图3　间断性应用场景

快速增长应用场景

快速增长应用场景指的是应用需求量增长比较迅速的情况。传统IT构建方式都是事先估计一个阶段的需求然后根据需求做系统容量配置。在这种方式下应用系统的容量扩展不是动态的，而是呈阶梯型上升的方式。对于快速增长的应用，传统方式会导致IT系统需要进行频繁的调整。而云计算可以通过动态分配资源的方式来满足应用快速增长的需要。如图4所示。

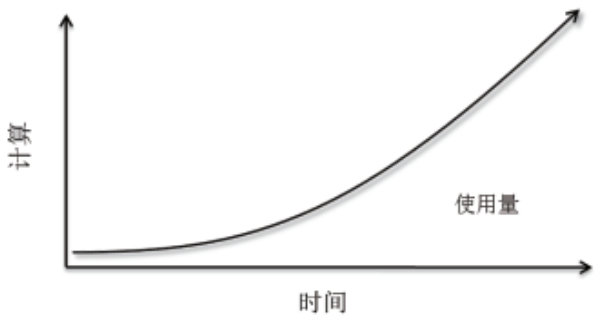


图4　快速增长应用场景

需求突增性应用场景

需求突增性应用场景是指应用的使用量短时间内快速增加有很快回落的情况。这种使用量的变化有的是无法预测的，比如当突发性事件发生时新闻网站访问量突增。也有一些是可以预测的，比如每年除夕的短信发送需求、节日的在线购物需求等。无论是哪种突发性需求对于应用系统的容量估算和保障都非常有挑战性，但是如果采用云计算就可以相对容易地解决这个问题。如图5所示。

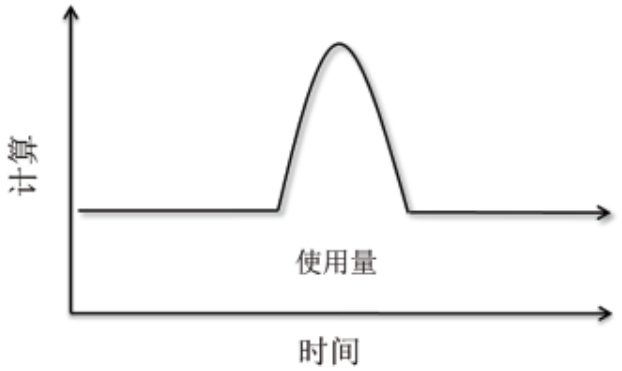


图5　突发性应用场景

资源利用率的分析

云计算能够为企业和用户降低成本，但是对不同规模企业采用不同类型云计算服务的时候，其成本优势是不一样的。云计算之所以能够降低IT服务的成本，最主要的原因是云计算服务的构建方式能够充分利用IT资源，并大量采用自动化的手段降低管理成本。另外相对于传统IT构建方式，云计算具有相当好的规模效应，这一点在公有云中表现得尤为明显。大型云计算服务如微软的Windows Azure、亚马逊的AWS等使用的服务器、存储、网络带宽，甚至是电力的单位成本都相比一般企业要便宜得多。如图6所示。

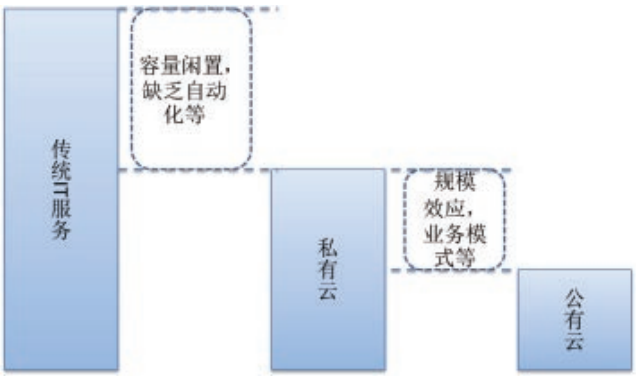


图6　云计算与传统IT服务的成本对比示意图

我们可以从图6看出，公有云的成本相对较低。但是对于那些不方便采用公有云的企业来说，私有云也可以帮助他们降低IT服务的成本。与采用公有云服务不同，私有云还是需要企业有大量的前期投资，但是它在资源利用率和管理成本上会比传统方式有比较大的进步。比如，私有云对企业的意义之一是对其现有IT硬件能力的充分发挥，根据调研大多

机构对服务器等IT资源的利用率一般都低于15%，但为了应对业务峰值的需要或预期负载的偏差，用户不得不闲置超过85%的IT能力。但是通过运用云计算的技术可以让用户实现均衡的、优化的硬件资源管理，从而使企业不再需要进行过度的投资。传统IT和私有云方式的成本比较如图7所示。

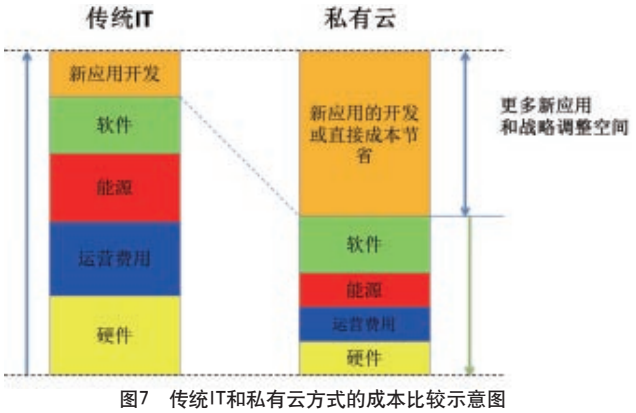


图7 传统IT和私有云方式的成本比较示意图

平台的资源需求分析

下面我们简单对传统IT和私有云两种不同方式对硬件的需求进行一下分析。如果我们把任意应用的资源需求用Appij表示，其中i为应用序号，j为任意时刻，那么所有应用的资源需求分布可以用一矩阵A表示如下：

$$A = \begin{pmatrix} App_{11}, App_{12}, \cdots, App_{1t} \\ App_{21}, App_{22}, \cdots, App_{2t} \\ \cdots \\ App_{n1}, App_{n2}, \cdots, App_{nt} \end{pmatrix}$$

在传统IT方式下，由于不同应用之间没有硬件的共享能力，因此对硬件的配置要求是所有应用系统高峰值的总和。我们可以用下面的公式来表示传统IT方式对硬件的需求：

$R = \sum\sum[Max(App_{1j}) + \cdots + Max(App_{nj})]$，其中j为任意时刻。

也就是说传统IT方式对硬件的需求为上面矩阵A中每一行最大值之和。

在私有云方式下，不同应用之间可以共享一个资源池，因此在一个理想的状况下，对硬件的需求是所有应用系统累加的高峰时刻。

可以用下面的公式来表示私有云方式对资源的需求：

$R = Max(A_1, A_2, \cdots, A_t)$，其中$A_j = \sum(App_{1j} + App_{2j} + \cdots + App_{nj})$

即矩阵A中的任意一列之和。也就是说私有云对硬件的需求为上面矩阵A中列和的最大值。

从上面两个公式的表达中我们可以推导出，在理论上私有云与传统IT方式的硬件资源需求在两种情况下是相同的。一个是当应用数量为1的时候，另一个是当所有N个应用的资源需求都是在同一时刻达到最大值（实际上前者也可以看成是N=1的特殊情况）。我们同时还可以看出，当不同应用之间的资源需求在时间上比较分散，这样从资源利用率方面私有云的效果会更加明显。而一般的当应用程序数量越多，其相互之间差异的可能性也就会相对高一些，从而显现出云计算的规模效应和共享的优势。

结束语

云计算被认为是自从发明分时系统和PC出现之后信息服务提供方式的一个最大变革。在过去的两年多时间内，云计算这个名词在厂商和媒体的推动之下已广为人知。云计算也正在给IT产业带来各种深远的影响。但是，对于不同的应用场景，云计算所提供的效果是有差别的，因此我们还是需要从实际需求出发来选择合适的方式。云计算是通过资源共享方式来提高资源的利用率，而资源集中是资源共享的前提。由于现实中的应用程序在使用资源的模式上往往具有互补的特性。比如，有些应用的高峰访问在白天，有一些是在晚上等。如果从全球配置的角度看，那样由于时差的存在，资源互补性就更为明显。因此，从资源利用率的角度来看这种资源集中供应的方式将使得资源分配达到最优化。也许不远的将来，整个互联网就像一台巨大的计算机，在它上面提供无限的计算资源和服务，人们使用它上面的应用程序就像我们现在使用自来水、电力那样方便和自然。

参考文献：

- Michael Armbrust, Armando Fox, Rean Griffith, etc., Above the Clouds: A Berkeley View of Cloud Computing, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf
- Peter Mell, Tim Grance, The NIST Definition of Cloud Computing, http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc
- Alexander Lenk, Markus Klems, Jens Nimis, Stefan Tai, Thomas Sandholm, What's Inside the Cloud? An Architectural Map of the Cloud Landscape, CLOUD'09, May 23, 2009, Vancouver, Canada

作者简介

方国伟，软件架构资深顾问，清华大学计算机系研究生，目前在微软开发工具及平台事业部负责微软云计算解决方案的技术推广工作。曾先后担任IBM（中国）有限公司软件部资深软件工程师、微软企业和合作伙伴部制造业客户技术经理。

■ 责任编辑：董世晓（dongsx@csdn.net）

基于反射的软件自动化测试框架设计(上)

■ 文 / 黄捷

文章以Windows SDK工具箱中Service Configuration Editor的实际测试为例，演示了反射在测试中的应用。

软件自动化测试的一种理想模式是软件开发人员为测试人员提供一组专供测试调用的接口，这组接口可以是产品正式代码的一部分（但是不会暴露给用户）。通过调用这组接口，软件测试人员可以模拟绝大部分的用户行为，从而很容易地编写测试代码。但是在实际工作中，因为种种原因，只有很少一部分的软件项目真正实现了这样的软件测试接口。这使得对于用户行为的模拟变得复杂了许多，特别是对于图形界面测试（以下简称UI测试）。另外，为了和产品代码的更新保持同步，测试代码的维护成本也相当之高。

反射（Reflection）技术的成熟使得软件测试人员在软件测试阶段可以轻松地得到软件内部几乎所有信息。虽然使用反射无法代替专门的测试接口带来的方便，但是在很多情况下，使用反射仍然能大大提高测试效率，节省维护成本。

Windows SDK工具箱中的Service Configuration Editor（以下称为服务配置编辑器）就是一个可以使用反射来帮助测试的好例子，是微软为Windows Communication Foundation（以下简称WCF）提供的一个配置文件编写器，是微软面向服务编程的主要编程模型。不过我们这里并不需要关心WCF的详细信息，只需要知道WCF的配置文件是一个XML文本。对于WCF初学者，了解这些设置并根据需要正确地进行配置颇有难度。 所以微软在Windows SDK工具箱里提供了服务配置编辑器，用以帮助和简化WCF的配置。

哪里可以找到服务配置编辑器

服务配置编辑器是Windows SDK工具箱的一部分，它包含在Windows SDK 6.0及以后的版本或Microsoft Visual Studio 2005及以后的版本中。以Visual Studio 2008 为例，你可以在如下位置找到它： “开始菜单->所有程序->Microsoft Windows SDK v6.0A->Tools->Service Configuration Editor”。图1显示了服务配置编辑器的主要界面：这个界面的左边显示了一棵节点树，对应着XML配置文件中的树结构。当你选中这个节点树中某一个带有可编辑属性的子结点，右边的面板就会把这些属性显示出来。你可以在这里编辑你想要改变的属性。这些改变会被保存到对应的XML配置文件中去。本文的内容将主要围绕这个图形界面的测试展开。

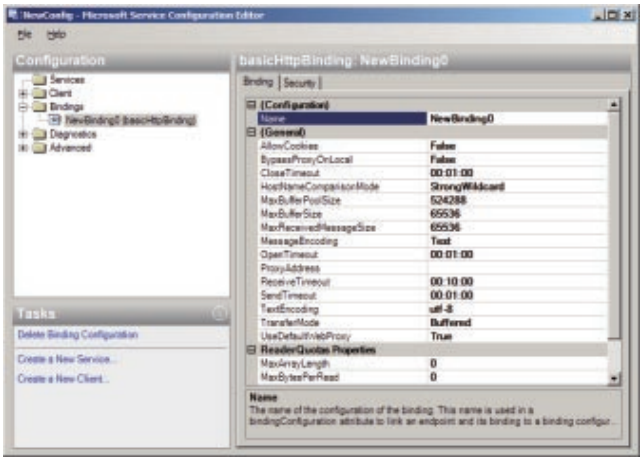


图1 服务配置编辑器的主要用户界面

反射简介

反射是在程序运行时，通过元数据（Metadata）或者其他手段得到程序内部的数据和非数据元素信息，修改或者操作其结构和行为的技术。如今很多流行的编程语言都提供对反射的支持，本文将以C#为例。具体实践中，使用和测试对象一致的编程语言，将会给反射带来最大的方便。在C#中，反射允许你获得加载到应用程序域（Application Domain）里面的程序集、类型、方法等信息。可以说通过反射，程序集内部的结构对调用者来说一目了然。.NET 的反射还可以用来动态地创建类型并且调用方法。下面是两个例子。

通过反射得到一个方法的信息：

```
// 要被反射的方法
public void MethodA() { }
static void Main(string[] args)
{
    // 用反射得到方法信息
    MethodInfo mInfo = typeof(Program).GetMethod("MethodA");
```

```
    Console.WriteLine("Found method: {0}", mInfo);
}
```

通过反射动态创建并调用一个方法：

```
//一般的方法调用
Foo foo = new Foo();
foo.Hello();
// 通过反射动态调用
Type t = Type.GetType("FooNamespace.Foo");
object foo = Activator.CreateInstance(t);
t.InvokeMember("Hello", BindingFlags.InvokeMethod,
  null, foo, null);
```

.NET的反射类型在System.Reflection名字空间下。

反射在UI测试中的应用

反射已经不是你需要在特殊情形下，或者已经走投无路时才应该考虑的技术了。很多情况下，虽然在产品代码里面使用反射是一个需要按实际情况讨论的问题，但在测试代码中，只要使用反射不会成为影响测试效率的瓶颈，就可以随时随地大胆地使用它。特别是当你测试的图形界面是底层Framework或者API的表现层，且在开发阶段随着底层框架的变化而一起变化时，尤其应该考虑使用基于反射的自动测试框架是否会给你带来意想不到的测试效率（测试代码可以依据用反射获得的底层框架信息来动态决定图形界面操作的步骤）。在较大的软件项目中，有时图形界面和底层基础架构的开发往往是由不同的团队承担，由于沟通或者地域差异等问题，底层的变化很可能无法及时地反映到图形界面中来。在这种情况下，使用基于反射的测试框架更能够在第一时间发现底层的变化，节省沟通成本（这在跨时区的协作开发中往往是一项很大的成本）。

服务配置编辑器的测试项目符合了以上所有特性。它是底层WCF框架中所有可配置属性的表现层，它的显示内容随着底层框架内容的改变而改变。服务配置编辑器的开发和测试独立于底层WCF框架的开发，分属于横跨十多个时区的多个团队（有多个团队分别负责开发WCF框架的不同部分，共同对服务配置编辑器产生影响）。如果每次WCF框架有任何微小变化，都需要测试工程师手动修改测试代码来达到同步（我们曾经这样做过），这样的维护成本是巨大且让人厌倦的。利用反射，只要整个产品框架没有大变化，维护工作都可以化繁为简，甚至化简为零。文章的后半部分将详细介绍如何做到这一点。

基于反射的服务配置编辑器测试框架设计

图2显示了基于反射的UI自动测试用例的基本执行流程。下面将讲解每个步骤。

测试用例文件的定义和读取

本例中，测试用例是用一个单独的定义文件来描述的，而不是直接在代码中定义。我们需要为测试用例定义文件规定一个合适的格式，实际上也就是建立测试用例文件和测试框架之间通信的协议。虽然看起来很简单，但是整个测试框架的设计思路在这一步完成时应该已经能够勾勒出一个大概的剪影了。比如测试框架应该提供多少灵活性——在一个测试用例里是否能测试多个测试目标、是否能够指定特定的数据源或者指定自定义的初始化过程。这些设计的方方面面都会在测试定义文件的格式中体现出来。所以，在规定格式前务必要做好充分的设计准备。

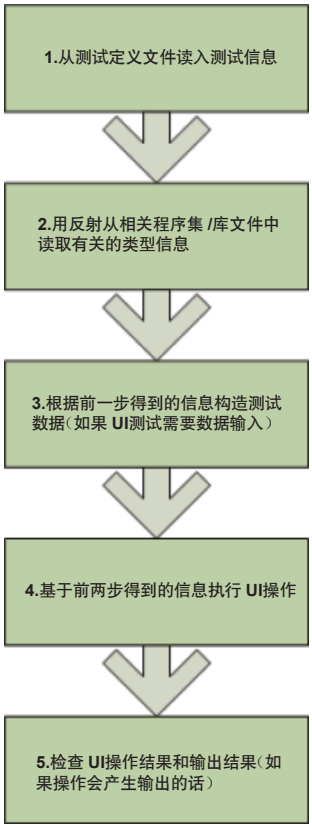


图2 基于反射的UI自动测试用例的基本流程

接口与数据分离：接口与数据分离在很多情况下都是需要考虑的设计注意点。首先，使用测试定义文件，而不是把测试信息直接定义在程序里本身就已经体现了接口和数据分离的思想（测试用例定义文件是数据，测试框架程序提供必要的接口从文件中读取测试信息）。其次，将测试定义文件的读入接口和读入后测试信息的存储形式分离。以面向对象的说法来说，就是分别为测试信息的读取和存储设计不同的类。这种分离可以带来诸多好处。比如，当项目进行到一半的时候，突然发现需要更改测试定义文件的格式，或者增加对另一种格式的测试定义文件的支持（这种项目进行中的突发事件有时候是难以避免的）。在这种情况下，如果没有在设计之初就注意到接口和数据的分离，那么这样的改动将会是非常痛苦且耗时的。相反，如果测试信息的读取类和读入程序以后数据的存储类是分开的，那么，只要修改或者添加一个读取类就可以适应这样的变动。接口与数据分离的好处有很多，比如增加项目的设计清晰度，帮助更好地分工协作，增加项目的灵活性和强健性。在后面的内容中还会提到，如何利用接口和数据的分离，帮助重现发现的软件问题。图3表示了测试用例文件读入和存储过程中的基本类结构。

服务配置编辑器的主要测试目标是WCF Service的XML配置文件中一些对应的元素和元素的可配置属性。图4中左边显示的NewBinding0(basicHttpBinding)节点对应的就是配置文件中的一个绑定元素（Binding），括号中的

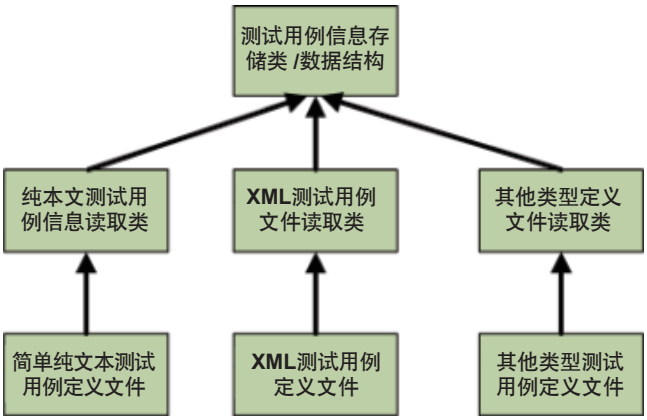


图3　测试用例读取的类结构

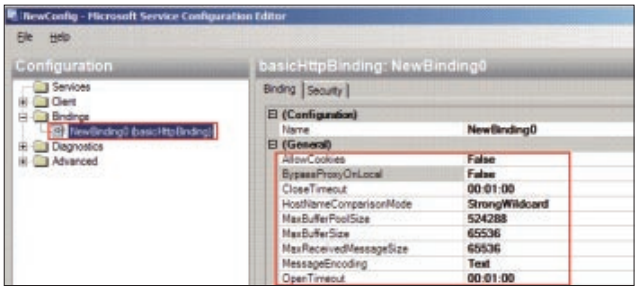


图4　服务配置编辑器测试目标示例

basicHttpBinding表示了这个绑定元素的类型。右边面板显示的是这个绑定元素的一些可配置属性。

服务配置编辑器的测试用例定义文件支持三种类型：一种是纯文本，一种是XML，还有一种用于问题重现，会在后面讨论。支持多种类型当然不是必须的，只是为了给不同需求的测试用例提供更多方便。例如，要测试图4中所示的basicHttpBinding类型的绑定元素，最简单的方法就是定义一个只包含元素类型名的纯文本文件：

```
ElementName : BasicHttpBindingElement
```

测试执行程序注意到这个文件中仅仅包含了一个元素名， 就知道用户希望所有的测试都按照默认设置行进。在默认的设置下，测试程序会对该图形界面元素进行尽可能全面的测试，包括新建、编辑属性、删除、错误提醒检查等。这样，创建一个普通测试用例就变得十分简单。

使用XML类型的测试定义文件能方便指定更多的用户选项。比如，我只想测试某个元素的某几个属性，或者我想在执行某个操作之前做一些初始化的工作。XML的清晰层级结构可以帮助我为每个测试目标量身定制测试的设置。这对于定义一个已发现问题的回归测试（Regression测试，防止已经修正的软件缺陷再次出现的测试）特别有用。最后，通过定义合适的XML结构定义文件（XSD文件）， 可以很轻松地在测试执行之前检查测试定义文件语法的正确性。

测试文件一旦读入，你可以用任意适合具体需求的类或数据结构来进行存储。只要以不同方式读取的测试信息是用同样的类或数据结构来保存就可以了。

用反射从相关程序集中获取类型信息

获取类型信息：在这一步中，我们首先要定义一个相关的程序集列表。这些程序集中应该包含所有我们需要反射的类型。测试项目应该添加这些程序集的引用（或者动态加载它们），然后在需要时在这个列表中搜索并反射类型信息。如果这个列表很长，并且类型搜索的次数很频繁，我们可以缓存每一次搜索的结果来提高类型搜索的效率。下面是C#中用反射加载程序集并且得到目标类型的一种方法：

```
// 从程序集的长名加载程序集
Assembly asm = Assembly.Load(
"system.servicemodel, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089");
// 从程序集中得到类型
Type t = asm.GetType("System.ServiceModel.
Configuration.BasicHttpBindingElement");
```

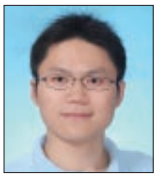
上面代码的缺点是，当程序集版本号升级时，测试代码也要随着一起修改。为了避免这一问题，可以从程序集中一个典型类型出发获得该程序集。这样，只要这个类型的名字不变，我们就不需要更改测试代码。请看下面的示例：

```
// 得到包含指定类型的程序集
Assembly asm =  Assembly.GetAssembly(typeof(System.
ServiceModel.Configuration.ServiceElement));
// 从程序集中得到类型
Type t = asm.GetType("System.ServiceModel.
Configuration.BasicHttpBindingElement");
```

获取可配置属性信息：得到了想要的类型之后，这个类型的所有信息再反射下就一目了然了。对于服务配置编辑器来说，我们需要通过类型来得到它所有的可配置选项，从而对这些选项进行测试。所有的可配置选项，都是通过一个名为“ConfigurationProperty”的属性（Attribute）来标明的。

很多情况下，目标元素还包含子元素（像XML配置文件中的子节点一样）。在这种情况下，我们需要递归地去反射所有后代元素的可配置属性，并最终得到一棵元素和可配置选项的树。这棵树就对应了一个目标元素的所有需要被测试的可配置属性。我们最好建立一个类或者数据结构来存储这些测试信息，并作为之后的测试步骤的起点。

作者简介



黄捷，2008年至2010年8月就职于微软亚太研发集团服务器与开发工具事业部，担任软件开发测试工程师，参与开发Visual Studio 及.NET Framework 相关工具。目前就读于哥伦比亚大学的软件工程硕士学位课程。

■ 责任编辑：董世晓（dongsx@csdn.net）

OPhone 3D应用开发之阴影渲染

■ 文 / 薛永

文章介绍了如何使用OPhone提供的3D API实现平面阴影渲染，以期能引领更多的开发者步入神奇的3D世界。

在3D世界中，为了增强场景的真实感与层次感，阴影处理非常重要。阴影是景物空间中光源不能直接照射到的区域，因此，要做出正确的阴影效果，需要对整个场景进行处理，并判断哪些物体会投射到哪些位置，即便对于飞速发展的PC 3D硬件来说，阴影处理也是一项非常繁重复杂的任务，而对于当前硬件设备和资源都极其有限的移动设备来说，只能通过一些特殊的方式来模拟实现阴影效果。

平面阴影原理

平面阴影处理是将阴影看成物体投射到某个平面的表面，也就是类似于将物体模型紧压在接受阴影的平面上，使其变成一个没有厚度的阴影区域。如图1所示，光源将物体的阴影投射到地面的表面，形成黑色的阴影部分。需要注意的是，如果地面有凹凸不平的起伏，平面阴影将无法适用，这也是该方法最大的限制。

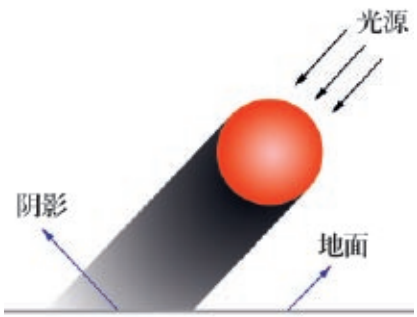


图1 平面阴影原理

要模拟实现平面阴影，需要得知：光源方向和位置，通常为平行光或者点光源；接受阴影的平面，通常由平面法线和平面上某一点定义而成。

通过以上条件，可以最终计算得到一个物体阴影变换矩阵（推导过程参见http://www.devmaster.net/articles/shadowprojection/）。使用时，将阴影变换矩阵与物体的当前模型视图矩阵相乘，最终得到将物体“压扁”到该平面的模拟阴影效果。

计算平面阴影变换矩阵

如图2所示，当3D场景中没有阴影的时候，模型和地面的接触会让人感觉失真，缺乏立体感。我们要做的，就是将3D模型“压扁”投影到地面上，并以黑色阴影色将压扁后的模型数据渲染到模型与地面接缝处来实现平面阴影效果。

我们将人物放置在世界坐标（1，0，1）处，将光源设置在人物头顶正上方（1，100，1）处，而用于接收阴影投射到地面（即图2中的红白方格地面），我们定义平面法线为竖直向上（0，1，0）。最终计算的平面阴影矩阵保存在mMatShadow中，供渲染时使用。程序中的相关定义如下：

```
/*阴影平面法线*/
private Vector3f
mvNormal = new
Vector3f(0, 1, 0);
/*阴影平面定位点*/
private Vector3f
  mvPoint = new
  Vector3f(1, 0, 1);
/*阴影平面*/
private Vector4f
  mvShadowPlane = new
  Vector4f();
/*光照位置*/
private Vector4f
  mvLightPosition = new
  Vector4f(1.0f,
100.0f, 1.0f, 0.0f);
/*最终的阴影矩阵*/
private Matrix4f
  mMatShadow = new
  Matrix4f();
```

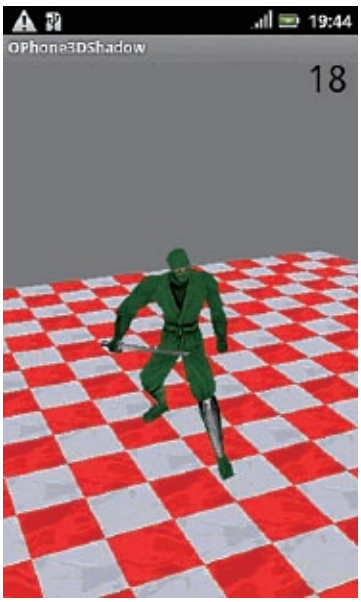


图2 不带阴影的3D场景

之后我们通过调用computeShadowMatrix()来计算得到人物模型的平面阴影矩阵以供后边渲染阴影时使用。得到平面阴影变换矩阵之后，便可以在渲染时将阴影矩阵与当前人物模型视图矩阵相乘以渲染出最终的阴影效果。

渲染阴影

在没有渲染阴影时，我们的渲染次序是先渲染地面，再渲染模型；而阴影是位于地面和模型之间的一个没有厚度的扁平面，因此渲染次序变为：

```
// 渲染地面
mGround.draw(gl);
// 渲染阴影
```

```
drawShadow(gl);
// 渲染模型
mModel.render(gl);
```

由于阴影只是或浓或淡的黑色区域，因此渲染阴影时，需要禁用相关纹理操作，并以纯色渲染。同时由于阴影往往与地面贴得极近，如果开启深度测试，则很容易造成深度冲突导致渲染时阴影和地面相互交错或者闪烁冲突，因此，在这个演示程序里，我们在渲染阴影时禁用了深度测试，同时关闭了深度写入，让之后的渲染物体可以覆盖阴影表面。我们将这些阴影渲染前后的操作封装如下：

```
public static void glBeginDrawShadow(GL10 gl) {
    //禁用深度测试
    gl.glDisable(GL10.GL_DEPTH_TEST);
    //禁止深度写入
    gl.glDepthMask(false);
    //禁用纹理贴图
    gl.glDisable(GL10.GL_TEXTURE_2D);
    //启用颜色混合
    gl.glEnable(GL10.GL_BLEND);;
    //设置阴影颜色为黑色不透明
    gl.glColor4f(0.0f, 0.0f, 0.0f, 1.0f);
}
public static void glEndDrawShadow(GL10 gl) {
    //启用深度测试
    gl.glEnable(GL10.GL_DEPTH_TEST);
    //启用深度写入
    gl.glDepthMask(true);
    //禁用颜色混合
    gl.glDisable(GL10.GL_BLEND);
}
```

我们在MS3D模型类中添加renderShadow()函数，并传入之前计算得到的阴影变换矩阵用于最终的渲染阴影，相关代码如下：

```
public void renderShadow(GL10 gl, Matrix4f matShadow) {
    gl.glPushMatrix();
    {
        //将阴影矩阵与当前模型视图矩阵相乘
        gl.glMultMatrixf(matShadow.asFloatBuffer());
        //开启阴影渲染模式
        ShadowUtil.glBeginDrawShadow(gl);
        //仅启用顶点位置数据
        gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
        // 遍历所有的MS3D Group，渲染每一个Group
        for (int i = 0; i < mpGroups.length; i++) {
            if (mpGroups[i].getTriangleCount() == 0) {
                // 如果该Group包含的三角形个数为零，则直接跳过
                continue;
            }
            //绑定顶点数据
            gl.glVertexPointer(3, GL10.GL_FLOAT, 0,
              mpBufVertices[i]);
            //提交渲染
            gl.glDrawArrays(GL10.GL_TRIANGLES, 0,
              mpGroups[i].getTriangleCount() * 3);
        }
        //重置相关操作
        gl.glDisableClientState(GL10.GL_VERTEX_ARRAY);
        //停止阴影渲染
        ShadowUtil.glEndDrawShadow(gl);
    }
    gl.glPopMatrix();
}
```

在上面的代码中，我们首先将阴影变换矩阵通过调用glMultMatrix()来与当前的模型视图矩阵相乘，来得到最终的变换矩阵。之后开启阴影渲染模式，并按照MS3D的模型渲染格式来将当前最新的模型渲染数据提交渲染。渲染完毕后，通知关闭阴影渲染模式，并重置相关状态。

最终效果

添加平面阴影渲染之后，最终的效果如图3所示。可以看到，阴影添加之后，随着人物模型动作的变换阴影也会实时地变化，这样大大增强了整个3D场景的真实感和层次感，并且对于这种平面的地面效果非常完美。大家可以通过上下左右拖动镜头来体验不同角度下的阴影表现。

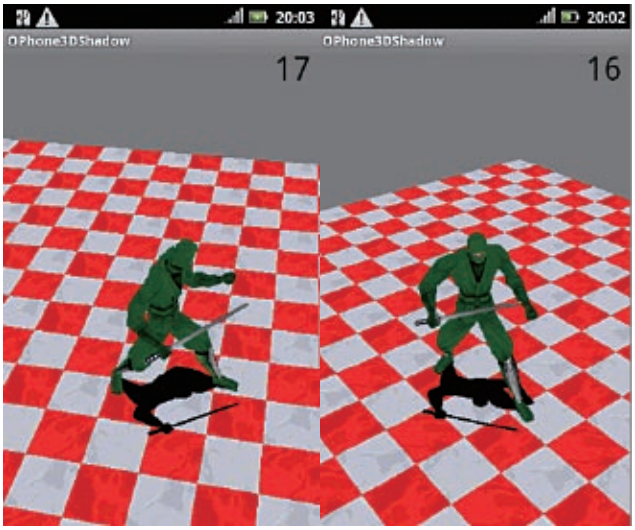


图3　添加阴影效果后的渲染图

总结

本文通过介绍如何使用OPhone中提供的3D API来实现平面阴影渲染，希望能引领更多开发者步入神奇的3D世界。随着OPhone 2.0平台的发布和推广，移动设备的3D能力必将越来越强大。

作者简介

薛永，软件工程师，专注于移动3D技术开发，目前正在完善跨PC/iPhone/Android NDK的统一3D引擎。

■ 责任编辑：董世晓（dongsx@csdn.net）

转储分析之探寻唤醒失败原因（下）

主持人：张银奎

《软件调试》一书作者，从事软件开发和研究10余年，对IA-32架构、操作系统内核、虚拟技术，尤其对软件调试有较深入的研究。翻译（合译）作品包括《数据挖掘原理》、《机器学习》、《人工智能：复杂问题求解的结构和策略》等。

又到了桂花飘香的季节，每当闻到那沁人心脾的香气，我都会心旷神怡，感叹大自然的神奇造化。我家楼下刚好也有几棵桂花树，正盛开着金色的桂花，刚才从旁边走过时，我故意深呼吸几次，希望能多吸入一些香气，然后通过文字把这迷人的芳香传递给各位读者朋友。上一期分析了系统挂死在唤醒途中的案例，通过!locks命令找到了执行睡眠和唤醒任务的关键线程，发现这个线程陷入等待了，在线程的栈回溯中，USB转串口设备的驱动程序位列其中，本期将继续探索其中的来龙去脉。

解析参数

仔细观察上一期找到的关键线程的栈回溯（图1），会发现其中存在明显的规律：#3、#4号栈帧对应的是ser2pl驱动的函数，其下是Serenum驱动的函数（#8、#9），再往下又是ser2pl驱动的函数（#e），而后又是Serenum驱动的函数（#12、#13）。而且两次出现的Serenum驱动的函数是一样的，都是Serenum_FDO_Power和Serenum_Power，两个驱动之间都有系统的PoCallDriver函数。为什么反复调用这两个驱动呢，是否存在死循环呢？要回答这个问题，就需要知道每次调用的原因是什么。

我们既没有驱动程序的源代码，又没有操作系统的源代码，如何进一步了解栈回溯中函数调用的细节呢？一条基本的经验就是先寻找自己熟悉的函数，然后根据熟悉的函数推测未知函数。比如，在图1的诸多函数中，PoCallDriver函数是文档化了公开函数，很容易可以找到它的原型：

```
NTSTATUS  PoCallDriver(
  IN PDEVICE_OBJECT  DeviceObject,
  IN OUT PIRP  Irp );
```

图1　执行睡眠和唤醒任务的关键线程

也就是说，PoCallDriver的第一个参数是设备对象，第二个参数是IO请求包（IRP），设备对象间接指定了要调用的驱动，IRP指定了需要处理的任务，即本次调用的具体原因。因此可以从栈回溯中找到PoCallDriver一行的第一个参数取值，然后使用!devobj命令观察调用的是哪个驱动，比如对于#11栈帧，只要执行!devobj 85163500（图2）。可见，85163500是ser2pl驱动的设备对象，即：栈帧#11的PoCallDriver是SerEnum驱动在调用Ser2pl驱动，栈帧#0e认证了这一点。可以知道整个栈回溯中：

- #16号栈帧是在调用Serenum驱动；
- #11号栈帧是在调用Ser2pl驱动；
- #0c号栈帧又是在调用Serenum驱动；
- #07号栈帧又是在调用Ser2pl驱动。

```
0: kd> !devobj 85163500
Device object (85163500) is for:
 Serial2 \Driver\Ser2pl DriverObject 86851358
Current Irp 00000000 RefCount 1 Type 0000001b Flags 0000204c
Dacl e187322c DevExt 851635b8 DevObjExt 85163e70
ExtensionFlags (0000000000)
AttachedDevice (Upper) 8610d2b0 \Driver\serenum
AttachedTo (Lower) 85164748 \Driver\ACPI
Device queue is not busy.
```

图2　观察设备对象

下面再看调用原因，也就是IRP。按道理PoCallDriver的第二个参数就是IRP，用!irp命令观察第二个参数就行了。但是这样做会得到一个错误，以#11为例：

```
0: kd> !irp 85163e70
IRP signature does not match, probably not an IRP
```

这说明这样读取到的数值并非是指向IRP结构的指针。这是什么原因呢？看一下PoCallDriver函数的反汇编就可以得到答案，在这个函数的开头处，在把Irp参数赋给寄存器后，便把另一个值写到了Irp参数的位置：

```
mov dword ptr [ebp+0Ch],eax
```

也就是说，PoCallDriver函数在接收到Irp参数后，将这个参数的位置赋成了其他值。于是我们只能另外想办法找正确的Irp参数值了。根据经验，PopPresentIrp函数的第二个参数也是指向IRP结构的指针，以#10为例，执行!irp 8761f600便可以得到IRP的详细情况。如图3所示，方括号开始的每个区块代表一个IRP栈位（Stack Location），对应于设备栈中的一个驱动。括号中的数字分别代表IRP请求的主编号和子编号，在头文件wdm.h可以找到它们的定义，比如[16, 2]分别对应IRP_MJ_POWER和IRP_MN_SET_POWER。

```
// wdm.h
#define IRP_MJ_POWER                    0x16
#define IRP_MN_SET_POWER                0x02
```

```
0: kd> !irp 8761f600
Irp is active with 6 stacks 5 is current (= 0x8761f700)
No Mdl: No System Buffer: Thread 00000000:  Irp stack trace.
    cmd  flg cl Device   File     Completion-Context
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
>[ 16, 2]   0  0 85163500 00000000 00000000-00000000
           \Driver\Ser2pl
              Args: 00000000 00000000 00000001 00000002
[ 16, 2]   0 e0 8610d2b0 00000000 80653cb0-865caac0 Success Error Canc
           \Driver\serenum     nt!PopCompleteSystemPowerIrp
```

图3　观察IRP

系统发送IRP时，总是先发给设备栈的最顶层驱动，顶层驱动再依次向下转发。

据此观察上面的栈 回溯，可以看到一共有两个IRP出现，从下向上第一个是8761f600（图1蓝色框），第二个是85879c60（图1绿色框）。再用!irp命令观察另一个IRP（图4）。

比较图3和图4中的两个IRP，可以看到请求编号都是IRP_MJ_POWER和IRP_MN_SE0T_POWER。有不同吗？

```
0: kd> !irp 85879c60
Irp is active with 8 stacks 5 is current (= 0x85879d60)
No Mdl: No System Buffer: Thread 00000000:  Irp stack trace.
    cmd  flg cl Device   File     Completion-Context
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 00000000 00000000 00000000 00000000
[ 16, 2]   0 e1 85be8b88 00000000 8c789c0c-85163ad4 Success Error Cancel pending
           \Driver\usbhub\ser2pl
              Args: 00000003 00000001 00000001 00000002
>[ 16, 2]   0 e0 85163500 00000000 b7aa8bb6-00000000 Success Error Cancel
           \Driver\Ser2pl\serenum!Serenum_FDOPowerComplete
              Args: 00000003 00000001 00000001 00000002
[ 16, 2]   0 e0 8610d2b0 00000000 80526a0c-00000000 Success Error Cancel
           \Driver\serenum     nt!PopCompleteRequestIrp
              Args: 00000003 00000001 00000001 00000002
[  0, 0]   0  0 8610d2b0 00000000 00000000-00000000
           \Driver\serenum
              Args: 85be8b88 00000002 00000001 00000000
[  0, 0]   0  0 00000000 00000000 00000000-00000000

            Args: 857640e8 80563340 85879c60 00000000
```

图4　观察另一个IRP

两种IRP

为了搞清楚上面两个IRP的差异，我们需要进一步分析IRP中的IO_STACK_LOCATION子结构，我们知道每个IRP包含多个IO_STACK_LOCATION子结构，其中有一个是当前驱动使用的，即驱动程序中经常通过如下语句取得的：

```
irpStack = IoGetCurrentIrpStackLocation(Irp);
```

执行!irp 85879c60 1命令，显示更加详细的IRP信息（省略），注意其中的CurrentStackLocation字段：

```
Tail.Overlay.CurrentStackLocation = 85879d60
```

CurrentStackLocation是一个_IO_STACK_LOCATION结构，可以使用dt命令观察：

```
0: kd> dt _IO_STACK_LOCATION 85879d60
nt!_IO_STACK_LOCATION
   +0x000 MajorFunction    : 0x16 ''
   +0x001 MinorFunction    : 0x2 ''
   +0x002 Flags            : 0 ''
   +0x003 Control          : 0xe0 ''
   +0x004 Parameters       : __unnamed
```

其中的Parameters字段是一个联合结构，对于我们目前的电源类IRP，它是一个Power子结构：

```
struct {
  ULONG SystemContext;
  POWER_STATE_TYPE Type;
  POWER_STATE State;
  POWER_ACTION ShutdownType;
} Power;
```

其中POWER_STATE_TYPE、POWER_STATE和POWER_ACTION都是枚举常量，可以用dt命令显示其定义：

```
0: kd> dt _POWER_STATE_TYPE
nt!_POWER_STATE_TYPE
   SystemPowerState = 0
   DevicePowerState = 1
```

因此，Power结构实际上就是包含四个整数，我们可以直接用dd命令观察它：

```
0: kd> dd 85879d60+4 l4
85879d64  00000003 00000001 00000001 00000002
```

也就是说，SystemContext（系统保留）的值为3；Type的值为1，代表DevicePowerState，即设备电源状态；State值为1，代表PowerDeviceD0，即正常工作状态（D0）；

ShutdownType的值为2，代表PowerActionSleep。至此，我们知道85879c60这个IRP是用于设置设备电源状态的。

针对另一个IRP 8761f600，重复上面两个步骤，可以得到它的Power结构为：

```
0: kd> dd 8761f700+4 l4
8761f704  00000000 00000000 00000001 00000002
```

即SystemContext的值为0；Type的值为0，代表SystemPowerState，即系统电源状态；State值为1，代表PowerSystemWorking，即正常工作状态（S0）；ShutdownType的值为2，代表PowerActionSleep，看来，这个IRP是用于设定（通知）系统电源状态（变化）的。

这两个IRP有不同用途，一个用于通知系统电源状态变化，另一个用于设置设备电源状态变化。可以作出如下推论：

- 系统唤醒过程中，系统函数PopWakeDeviceList向USB转串口设备发送唤醒通知，具体说就是向设备栈的最上端设备对象8610d2b0（Serenum）发送代表系统电源变化的IRP；
- Serenum驱动将IRP调用向下传递给Ser2pl驱动；
- Ser2pl驱动作为负责设备电源状态的FDO，发现电源状态（D0）和当前状态（D3）不一致，于是调用PoRequestPowerIrp发起IRP调用，企图改变设备的电源状态。

这样看来，线程中并没有发生死循环。

为何等待

弄清了两轮IRP调用后，我们继续分析当前线程为何进入等待（栈帧#02），表面上看，就是因为这个等待而导致了唤醒过程阻塞不前。集中观察靠近栈顶的几次函数调用：

```
0: kd> kn
*** Stack trace for last set context - .thread/.cxr
resets it
 # ChildEBP RetAddr
00 b6c50340 80503846 nt!KiSwapContext+0x2f
01 b6c5034c 804fb078 nt!KiSwapThread+0x8a
02 b6c50374 8c78b51d nt!KeWaitForSingleObject+0x1c2
03 b6c503a4 8c78b83f ser2pl+0x751d
04 b6c503c8 804ef19f ser2pl+0x783f
```

看了这个栈回溯，很容易想到#03号栈帧为什么要调用KeWaitForSingleObject函数呢？虽然没有ser2pl驱动的源代码和符号文件，但通过反汇编分析了一会#03号栈帧所对应的函数后，我意识到这个函数与DDK中的串口驱动源代码非常相似，图5给出了来自DDK（版本号3790）的有关代码片断。

```
KeClearEvent(&pDevExt->PowerD0Event);


IoCopyCurrentIrpStackLocationToNext(Pirp);
IoSetCompletionRoutine(Pirp, SerialSyncCompletion, &pDevExt->PowerD0Event,
                       TRUE, TRUE, TRUE);

SerialDbgPrintEx(SERPMPPOWER, "Calling next driver\n");

status = PoCallDriver(pDevExt->LowerDeviceObject, Pirp);

if (status == STATUS_PENDING) {
   SerialDbgPrintEx(SERPMPPOWER, "Waiting for next driver\n");
   KeWaitForSingleObject (&pDevExt->PowerD0Event, Executive, KernelMode,
                          FALSE, NULL);
```

图5　进入等待的有关代码

图5中代码的基本过程是，先复位PowerD0Event，再将当前的IRP栈位复制一份到下一个栈位。第3行（不含空行）是设置一个完成函数SerialSyncCompletion，将指向PowerD0Event的指针作为完成函数的参数。第6行是调用PoCallDriver将IRP传递给设备栈中更底层的驱动。栈上为我们提供了一个很好的证据，观察图1中#3号栈帧第二个参数的位置，其值为00000103，它就是STATUS_PENDING常量的值，从汇编代码可以看到，在PoCallDriver返回后，有一条MOV指令将EAX中的返回值赋值到EBP+C的位置，即第二个参数的位置：

```
mov dword ptr [ebp+0Ch],eax
```

分析到这里，我们清楚了唤醒线程进入等待的原因，是Ser2pl驱动收到设置系统电源的IRP后，发起设备电源状态变化的IRP，而后它在收到和处理设备电源变化IRP时，在把IRP传给其下的驱动时，PoCallDriver返回STATUS_PENDING，因此Ser2pl调用等待函数进入等待。

等到何时

图5中的等待函数何时结束返回呢？因为最后一个参数Timeout指定为NULL，即永远等待，所以只有等待成功，函数才会返回，也就是用要有人设置等待的PowerD0Event。哪里设置这个Event呢？完成函数SerialSyncCompletion中会设置：

```
NTSTATUS
SerialSyncCompletion(IN PDEVICE_OBJECT DeviceObject,
IN PIRP Irp, IN PKEVENT SerialSyncEvent)
{
   KeSetEvent(SerialSyncEvent, IO_NO_INCREMENT,
FALSE);
   return STATUS_MORE_PROCESSING_REQUIRED;
}
```

谁会调用这个完成函数呢？当前线程已经挂起，只能寄希望于其他线程。进入等待前，PoCallDriver已将IRP传递给底层驱动，顺利的话底层驱动以异步方式来完成这个IRP，完成的过程中，会调用完成函数，于是堵塞便化解了，而现在没有人来完成这个IRP，导致唤醒过程停滞不前，屏幕无法点亮。是什么原因导致IRP不能完成呢？后续文章将继续讨论。

本期问题：

上期的答案是WinLogon的唤醒线程在等待事件对象85163ad4（上期图4中可以找到答案）。本期的问题是!irp命令的结果中，Args后的四个整数的含义该如何解释，比如本期图4中的Args: 00000003 00000001 00000001 00000002是什么意思？

编者说明：

- 投稿信箱：contest@csdn.net
- 解答提交时间，最好能早于当月15日
- 联系方式写在一个单独的TXT文件里，包括以下几项：
  1）姓名　2）工作单位或学校　3）电话联系方式
  4）邮寄地址　5）E-mail地址

■ 责任编辑：董世晓（dongsx@csdn.net）

行进中的淘宝前端类库：KISSY

■ 文 / 王保平

书写干净漂亮的代码非常不易。作为前端工程师，还得处理恼人的浏览器兼容性问题。Protoype、jQuery、YUI、 MooTools等各种JavaScript类库，都是为了让前端工作更轻松自如。下面将介绍淘宝前端类库KISSY的起因、设计原则和核心功能，以及相关开发流程和社区建设。

淘宝之所以要自主研发前端类库和自身的发展历程息息相关。2007年开始，随着淘宝业务的快速发展，页面的数量和复杂性都在迅速增加。淘宝UED开始有专职的前端工程师。面对繁重的业务需求，寻求一个高效稳定的JavaScript类库非常重要。淘宝选择了YUI类库。YUI为淘宝的前端发展立下了汗马功劳。

从2007年下半年开始，淘宝的前端工程师发现，页面上的不少通用功能，都不能直接套用YUI的现有组件来完成。比如图片轮播、弹出层等，如果直接用YUI的组件，会导致要加载的文件很大，修改成本还不低。于是淘宝开始基于YUI的核心功能，实现了一套常用组件，加上一些实用的辅助工具类，形成了淘宝的第一代前端类库：TBra：由YUI Core和SimpleSlide等实用组件组成。到2008年初已趋于稳定。

2008~2009年，淘宝的业务更加迅猛增长。这两年里，jQuery类库风靡全球。YUI2是个优秀的类库，但在全球对“Write less, do more”渴望中，YUI2的传统API风格越来越不受前端工程师待见。

2009年6月份，YUI团队发布了YUI3 beta版本。YUI3借鉴了jQuery的不少API设计，从整体架构到代码实现，与YUI2有很大不同。YUI3激荡人心，但试用下来却让人感觉沮丧：虽然是颗粒化的，可按需加载，但常用功能打包后依旧偏大；9月份发布正式版后，稳定性上也并不如意。

2009年底，面对淘宝旺盛的业务需求，面对工程师对jQuery的认可，面对YUI2的落伍与缓慢更新，面对YUI3的诸多不如意，所有问题交织汇集在一起，淘宝前端面临着以下几条路：

- 继续使用YUI2，完善TBra；
- 全面升级到YUI3，包括TBra；
- 全面迁移到jQuery；
- 混用YUI2、YUI3和jQuery；
- 基于jQuery开发组件类库；
- 自主开发全新类库。

所有这些选择，在方向上，没有孰对孰错。TBra的设计初衷是快速、简单、易用，但在代码组织和可扩展性上，已逐步暴露出局限性。综合各种考虑，最后筛选下只剩下两条路：

- 基于jQuery开发组件类库；
- 自主开发全新类库。

以上是淘宝选择自主研发前端类库KISSY的心路历程。

设计原则

在设计领域，KISS代表简约原则：“Keep It Simple, Stupid”。KISSY的命名和设计原则皆源于此，总结为一句话是：小巧灵活、简洁实用，使用起来让用户感觉愉悦。

小巧是相对小巧。考虑国内的网络状况，要让用户快速打开页面，并不是一件容易的事。对于前端类库来说，小巧的体积非常有必要。jQuery的体积可以接受，YUI3在这一点上则处于劣势。KISSY会在功能和体积上权衡，尽量小巧。

灵活是适度灵活。KISSY核心功能的取舍，KISSY组件的特性选择，所有一切，都会尽量从淘宝的实际需求出发，在总结的通用使用场景下进行设计。

简洁实用是基本要求。一个类库，并非功能越强大越好。对于KISSY来说，保持API的简单易用，保持功能的精简实用，不求全但求精，力求用短小的代码完成常用的80%的功能。

使用起来感觉愉悦，这是更高层

次的追求。我们期望在使用KISSY时，你不会感觉枯燥乏味，而是在和一位老朋友愉悦交流。

我们还希望基于KISSY搭建的网站使得用户获得非常好的浏览体验和使用体验。KISSY的不少组件内置了对可用性的考虑，简单如标签页切换的延迟，复杂如富文本编辑器的交互体验，所有这一些，默认就尽量让用户能更好地使用。

核心功能

KISSY从设计之初，就尝试从各个类库中存精去芜，核心功能如图1所示：

种子（Seed）是使用KISSY的入口。通过种子文件，你可以自由灵活地加载所需要的组件。seed-min.js就是KISSY的种子文件，gzip后大小不到5 KB。页面中仅需引入它，就可以自由加载KISSY的任何组件。

核心包括DOM、Event、Ajax、anim、node、data等一系列常用功能。这些功能绝大部分是各大前端类库的标配。KISSY借鉴了jQuery简明易用的API设计风格，同时采纳了YUI3的颗粒化代码组织方式。种子和核心是构建各种组件的基石，详细功能请参考KISSY的API文档。

组件包括工具类组件和可视化组件，是KISSY最重要的组成部分。组件的公共接口和配置都尽量保持了内在的一致性。能让你在学会使用KISSY的一个组件后马上推测出其他组件的使用方式。

KISSY的组件严格遵守适度灵活原则，在特定的场景下进行设计。任何抽象场景，都有不同的抽象层次。比如淘宝首页的图片轮播，至少有以下三种层次的抽象：

- 图片轮播是可轮播的多张图片；
- 图片轮播是可切换的一组内容；
- 图片轮播是对命令的响应。

第一种抽象，会让我们做出像TBra里的SimpleSide类似的组件，功能很纯粹单一。大部分站点用jQuery等代码实现的图片轮播，也是基于这种简单明了的抽象。

第二种抽象，会让我们思考更多。

图 1 KISSY核心功能示意图

比如图片轮播和标签页切换的关系，从这一层的抽象来讲，标签页和轮播图片本质上一样的，都是一组可切换的内容。在这一层抽象上，我们就可以实现Switchable组件，有了Switchable，所有可以抽象为可切换内容的可视化组件，都可以快速基于Switchable来实现，而不是全新写一个。在淘宝首页上，除了搜索提示，其他可视化组件都是Switchable组件，非常便捷。

第三种抽象，很抽象，但抽象层次越高，落地越难。基本上所有的交互组件都可以抽象成为对用户行为的响应。适度灵活原则，能避免我们掉入过度抽象的陷阱。除了Switchable等常用组件，KISSY里还专门为部分复杂组件打造了子品牌专区，目前有KISSY DPL（设计模式库）子品牌、KISSY Editor（富文本编辑器）子品牌和KISSY AJBridge（Flash相关功能）子品牌。详细内容，请关注我们的文档和社区。

图 2 KISSY开发流程图

开发流程

一个类库要持续发展，需要一套行之有效的流程来保障。KISSY的开发流程可以总结为图2：

KISSY组件的开发非常重视预研阶段。我们建议每一个开发成员，至少花一半时间在组件的预研上。预研可以保证针对场景进行设计，可以让最后产出的公共API能被大家一致认可，能符合KISSY设计的整体要求。开发阶段，我们推荐BDD（行为驱动开发）。良好的BDD习惯和代码评审机制，能让我们的代码可靠，团队成员也能在过程中得到技能成长。单元测试框架目前KISSY选择的是Jasmine，目前还处于摸索起步阶段。但有热情和信心，就不怕远。

社区建设

KISSY是完全开源的，目前放在github平台上管理。KISSY已正式发布1.1.5版，下一个版本是1.2，将于今年12月份正式发布，敬请期待。非常欢迎各位朋友的试用与建议。

作者简介

王保平（花名玉伯），崇尚简洁而不简单，相信付出才有收获。就职于淘宝网UED部，忙并快乐着。

责任编辑：高松 gaosong@csdn.net

Sphinx: LAMP架构的新成员

■ 文 / 李沫南

作为目前LAMP架构上最契合的全文检索系统，本文介绍了Sphinx检索系统在提升Web应用性能方面的一些用途。

经常接触Web开发的同仁耳熟能详的一个词LAMP（Linux + Apache + MySQL + PHP），几乎成为Web开发的标配。在数据量越来越大、全文检索等功能日益成为网站标配的今天，Sphinx已经成为LAMP架构中心的组成部分，用于负责全文检索和整个系统性能的提升，权且称之为LAMPS架构。

回顾信息系统的早期，每个应用按照自己的需求定义自己专有的数据格式。为了便于应用程序的数据集成，降低数据存储模块的开发成本、提升可靠性，提出了SQL规范，出现了SQL数据库。数据库系统的设计目标是实现数据共享、减少数据冗余、实现数据一致性和便于维护等，数据库系统的访问性能仅仅是需要考虑的方面之一。而Web系统则不然，很多应用为了速度是可以牺牲部分一致性的。因此，使用SQL数据库直接为Web应用提供数据，必然会带来性能损失。为了弥补这一点，人们又提出了“读写分离”等优化手段。但是，由于使用的仍然是数据库，还有太多额外的计算能力被浪费掉了。优化数据的读取性能，应该有更好的解决方案。

而由于数据量的日益庞大，信息系统管理的中心也从如何维护数据变成了如何利用数据。而全文检索是目前为止少数几个已经被成功验证过的、可以有效利用数据的方法之一。而Sphinx的出现，弥补了LAMP架构在管理、利用数据上的短板。同时，由于其为只读应用优化，在客观上起到了提升Web应用响应速度的作用。

Sphinx Search在提供全文检索功能的同时，不需要对现有的LAMP系统架构进行大幅度的调整，就可以提升系统性能的解决方案。Sphinx项目自2001年开发，目前已经有10个年头了。在最新的1.0 beta中，支持了实时索引和SphinxQL等功能。

Lucene VS. Sphinx

只要提起全文检索功能，总有人会问与Lucene相比如何如何。实际上，与Sphinx做对比的基于Lucene的产品是Nutch。单单就引擎本身来说，Sphinx比Lucene在以下方面有优势：

更高性能：在默认配置的情况下，建立索引的速度是Lucene的5~10倍；处理搜索请求的速度是Lucene的3倍。

简单易用：与PHP和MySQL紧密集成，作为一个独立的服务进程出现，不需要再进行二次开发；还提供了SphinxQL接口，可以作为假的MySQL服务器运行。

内建中文支持和多种数据源的支持：（Coreseek的版本提供）系统内置了中文分词法和Python的数据适配层，支持多种数据来源

分布式支持：Sphinx还提供了初步的分布式索引服务支持，在压力大的情况下，可以方便地将检索请求分布到多台机器上。在国外的生产系统中，已经有接近2Tb的全文索引在运行。

Sphinx提升网站性能的实践

我们假设现有一个基于LAMP架构Web 2.0应用，如何使用Sphinx来提升性能呢？或者说，在SQL数据库与Sphinx混合部署时，那些任务应该是数据库做，哪些是Sphinx来做呢？答：一切Sphinx能完成的工作。下面结合网站应用的常见功能，介绍Sphinx在其中的助益。

图1是典型的使用Sphinx搭建的，

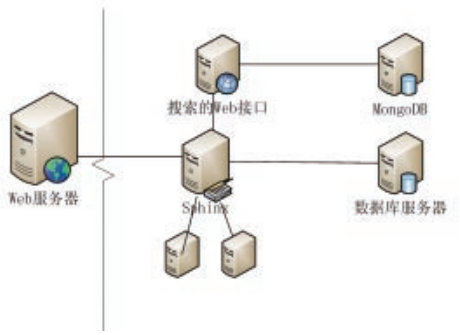


图 1 Sphinx全文检索网站系统示意图

基于全文检索的网站系统。其中，前台Web服务器的一切数据访问请求，都通过Sphinx全文检索引擎，由全文引擎返回查询结果。一般的，全文引擎从数据库中读取数据，建立索引。在少数无法预知数据模式的情况下，可以把数据放在MongoDB中，再交由Sphinx建立索引。

Sphinx系统支持初步的分布式处理，当单一的Sphinx节点及时处理检索请求时，可以通过数据分区、增加新服务节点等多种方式提升Web系统整体的请求处理能力。

结合具体应用的具体功能点，Sphinx可以在以下几个场景对Web系统提速。

列表页面：显示列表页面，可以使用Sphinx的Fullscan匹配模式，针对各种不同的检索需求，Sphinx提供了多种匹配模式，包括 Boolean、Phrase、Extended、Fullscan等。其中，如果用户没有输入查询关键词，并且属性的存储方式为extern时，自动切换到Fullscan模式。在Fullscan模式中，可以指定结果的排序方式，从而实现列表页面。

在默认的情况下，Sphinx可以支持前1000条记录，以每页20条记录计算，大约可以显示50页。

Tag云：很多应用都允许用户加标签，在基于数据库的应用中，往往还需要对标签做解析建立表关联……在Sphinx中，如果标签不允许用户自由输入，可以建立多值属性（MVA, multi-valued attributes），即一个属性名下可以挂多个值。如果允许用户自由输入，直接实现为一个全文索引字段即可。

此外，Sphinx还默认提供了对结果进行分组统计（聚类）的功能，可以对符合条件的记录进行计数，从而提供更易用的数据展现形式。

噪音过滤：在论坛应用或者类似应用中，经常会遇到引述其他人的文字。这些文字往往出现多次，对于全文索引造成噪音。而数据库或其他全文系统无法处理此种噪音，严重影响了检索结果的相关度。在Sphinx（Coreseek版本），由于用户可以使用Python脚本语言提供数据，可以在建立索引时单独对引用的文本或其他可能重复的文本进行检索或过滤，从而提升结果的相关度。

实时检索：对于频繁更新的内容，Sphinx支持主从索引结构。其中不常变化的放在主索引中，新增或修改的放在增量（从）索引中。在检索时，同时检索主从索引，并对结果进行排序。对于实时性要求特别高的应用，Sphinx还提供了RT Index功能。RT Index 本质上是一种内存Hash Table，为了保证系统的可靠性，Sphinx在操作内存的同时，还在磁盘上记录BinLog。当系统出现问题时，可以通过Log恢复索引。在未来版本，RT Index还将支持改写/另存为标准索引格式的功能。

无痛迁移：通过提供SphinxQL，Sphinx的服务端可以伪装自己是一台MySQL服务器，在1.0版本中，已经可以支持绝大多数常见的SQL语句。并且，如果要使用RT Index，SphinxQL是操作数据的唯一途径。可以把Sphinx视为一个功能有限，但是性能是普通MySQL服务器几倍的、一个专门为读取和全文检索优化过的MySQL服务器。

在国外的实践中，使用Sphinx来配合MySQL进行读写分离，可以提升4~10倍的系统处理能力。同时，由于增加了全文搜索等额外功能，产品设计人员可以更多地从用户的角度而不是从机器实现、数据库资源是否支持等方面设计产品。

进一步优化与功能限制

上面初步介绍了Sphinx检索系统在提升Web应用性能方面的一些用途，不过性能优化永无止境，还可以进一步地优化Web应用的性能。

字符串属性，现有的Sphinx仍然需要从数据库中再次读取数据，拼凑结果页面；1.0版本后，支持字符串属性。可以把显示结果摘要页面的数据直接存入字符串属性中，避免再次访问SQL数据库，进一步提升性能。

伪关键词，对于记录的过滤，既可以使用SetFilter等，也可以使用一个现实中不可能出现的词，放入全文索引中。使用伪关键词比使用属性值过滤可以再提速2倍左右。

定制排序，对于一个全文检索，而不是性能加速器来说，排序方式是否可定制非常重要。Sphinx支持多种权重计算方式默认是OKAPI BM25F，在得到权重值后，可以通过自定义排序公式调整结果的先后顺序，比如引入作者权重、点击数权重等。

当然，没有免费的午餐，Sphinx也存在若干缺点。比如：短语的ID使用CRC32或FNV64进行散列计算，存在一定的Hash冲突的可能；记录编号必须是整型等。然而，瑕不掩瑜，Sphinx依然是目前LAMP架构上最契合的全文检索系统。

对于LAMP应用或者说一切基于SQL数据库的Web应用，在决定花钱升级数据库或购买新硬件之前，不妨尝试一下Sphinx，或许会有惊喜。

作者简介

李沫南，系统分析师。863“Linux下兼容IE扩展的功能增强性浏览器”项目技术负责人。Coreseek.com 创始人。

责任编辑：高松 gaosong@csdn.net

新产品&新工具

Mac OS X Lion

10月20日，Apple 举办主题为“Back to Mac”的发布会。会上发布了名为Mac OS X Lion的操作系统，其创新功能包括：Launchpad苹果称之为“应用程序大本营”，相当于将iOS的Home屏设计引入到Mac OS X上，可以像在 iPad上一样排列所有应用程序，创建Folder文件夹来管理它们，你还可以滚屏操作；全屏应用程序——Mac全屏显示某个应用程序，可以更专注于当前的任务；Mission Control——任务控制，相当于之前的Expose功能，可以瞬间查看所有正在运行的程序，并且在它们之间切换。Mac OS X Lion定于明年夏天上市。

Mozilla官方发布项目：Kraken和Chromeless

名为“Chromeless”的实验项目，旨在帮助开发者简化浏览器界面开发，它试验性地移除现有的浏览器用户界面，用一个更有弹性的平台替代它，允许开发者使用标准的Web技术，如HTML、CSS和JavaScript创造新的浏览器界面。Firefox浏览器利用XUL实现多数的界面，但XUL还是存在着一定的限制，对完全定制造成了阻碍。目前源代码已经发布到github.com上。

浏览器基准测试工具Kraken，与Sunspider、V8和Dromaeo等测试工具不同的是，Kraken将重点放在现实性工作负荷和未来将出现的应用程序上。Mozilla相信这将能更好地促进浏览器发展。

微软发布Regez Fuzzer

Regez Fuzzer是一款免费工具，帮助程序员测试易受拒绝服务攻击的正则表达式。 由微软发布的SDL Regex Fuzzer，目的是测试程序员的正则表达式的执行条款。正则表达式模式中的某些条款（例如分组条款中包含重复自身的重复）执行起来极其耗时。这可能会被攻击者利用，发起拒绝服务攻击。该免费工具是一个受欢迎的程序员工具，尽管该漏洞只是受到拒绝服务攻击程序的一个方面。

jQuery Mobile 1.0 Alpha1发布

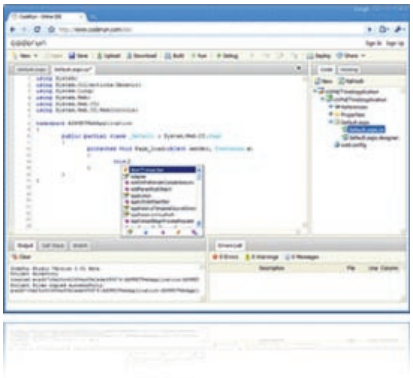
jQuery Mobile是jQuery在手机和平板设备上的版本。jQuery Mobile不仅会给主流移动平台带来jQuery核心库，而且会发布一个完整统一的jQuery移动UI框架。支持全球主流的移动平台。10月16日在波士顿举行的jQuery开发者会议上，jQuery开发团队正式发布了jQuery Mobile Project（jQuery移动项目）的首个alpha版本。这对于iOS、Android以及其他移动操作系统平台的开发者来说确实是一件值得高兴的事情。

MySQL的衍生版MariaDB

MariaDB是一个采用Maria存储引擎的MySQL分支版本，是由原来MySQL的作者Michael Widenius创办的公司所开发的免费开源数据库。最新版本是MariaDB 5.2.2。

基于Web的IDE开发工具：CodeRun

CodeRun是基于JavaScript语言、跨平台的集成开发环境，本身作为一款App产品，CodeRun 融入了分享机制，开发者可以有选择地上传项目代码，使用云技术来协同同事完成项目工作。目前，CodeRun主要支持一些Web开发语言，包括C#/.NET （3.5）、 PHP（5.1）、JavaScript、HTML以及CSS等，其中C#项目包括ASP.NET、WCF、Silverlight和WPF、MVC等项目，JavaScript脚本项目支持目前流行的JQuery、ExtJS、YUI等框架，其中数据库支持SQL Server 2005和Amazon SimpleDB。



phpMyFAQ

phpMyFAQ是一个多语言、完全数据库驱动的FAQ系统。它支持多种数据库来存储所有数据，需要 PHP4.1或更高的版本来访问数据。phpMyFAQ还提供一个带有WYSIWYG编辑器的内容管理系统，一个图片管理器、灵活的多用户支持、一个新闻系统、用户跟踪、一个模板系统、PDF支持、一个备份系统和一个易于使用的安装脚本。详细如下：修复XSS漏洞（建议所有用户升级）；增加对斯洛伐克语言的支持；改善后台界面；修复嵌入Google Analytics代码所引发的Bug。

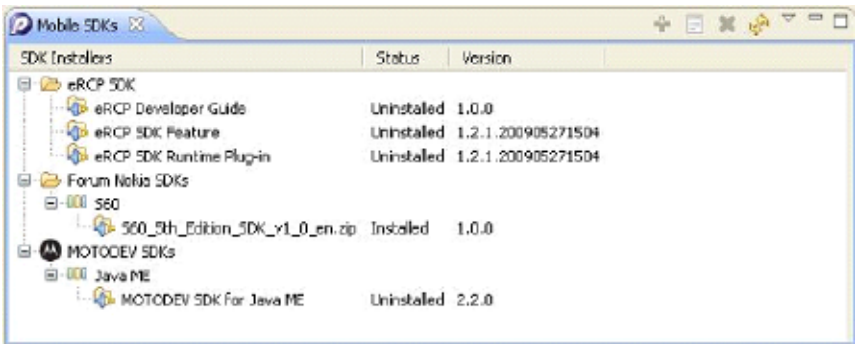
Ubuntu 10.10发布

代号为Maverick Meerkat的最新稳定版Ubuntu 10.10在10月正式发布，新版的Ubuntu 10.10在很多细节方面做了改变，比如新的主题，新的Sound Menu样式，软件中心（Software Center）做了大量改进，新的安装界面，新的Ubuntu字体，改进过的云服务Ubuntu One，由软件中心接管deb包安装过程等。另外，Ubuntu Netbook也启用了新的Unity界面。

eclipse pulsar

手机开发工具Eclipse Pulsar发布

Eclipse Pulsar 是一个基于 Eclipse 平台的手机应用程序开发工具。该项目由Eclipse成员Genutec、IBM、摩托罗拉、诺基亚、Research In Motion以及索爱共同发起。它是一个通用的工具，开发者无需再为各种不同的移动平台而下载不同的SDK。



BigTrak Jr

“向前5步，停下，右转30度，开火”——创造于1976年的可编程电动车BigTrak会记住并且依照顺序执行预先定义好的指令。它是一个六轮坦克车，有一个固定于前端的“蓝光”武器，背部有一个键盘用于输入指令。

GEEK产品

Logitech Revue

Google TV除了直接内置在电视机内部之外，还可以以外部设备的形式支持普通高清电视，Logitech Revue智能电视就是这样的一款产品。内置搜索和浏览器功能，支持HDMI高清接口，支持WiFi或有线连接。提供一个专用外部键盘作为输入设备（遥控器），也可以使用Android的智能手机作为遥控器。因为是基于Google TV开发，因此支持Android Market。300美元的价格，仅仅相当于一个上网本。

Brainovi

Brainovi是一款专为盲人设计的导航仪。同时具有语音提示和触摸屏显示功能，盲人能够依靠它而独立出行。使用时只需要对着导航仪说出目的地，导航仪上便会出现可触摸的3D地图，盲人通过触摸了解到路径。在行进过程中，辅助的蓝牙耳机会不断给出语音提示帮助盲人更方便地前进。有了它，可以让盲人更自由地行动。

SIEMENS miniTek

西门子的miniTek不但是助听设备的遥控器，还是连接无线通信设备的平台。相比传统的助听器，miniTek最大优势是能接收来自任何蓝牙设备的音频信息，然后充当耳机的角色把声音完整地还原出来。miniTek支持西门子现有所有的助听设备，如果和手机配合还能帮助听力障碍患者接听电话。和普通助听器一样，miniTek也能手动控制音量大小，并附带一个有线的音频输入插孔。

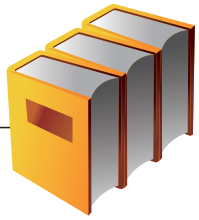
Razer Naga Epic

一向专注于游戏外设的Razer又发布了一款专为MMO游戏设计的鼠标。支持有线和无线两种模式，具有17个可编程功能的按键，精度达5600dpi，很多MMO游戏上都有对应的插件支持。内置电池，带充电插座，无线模式下可持续使用12个小时。

Sprint Overdrive

现在越来越多的电子设备都支持WiFi，但热点却不是随处可用的。虽然目前已经有了3G网络，但是大多数电子设备并不支持。Sprint推出的便携式3G路由器Overdrive可以利用3G网络组建无线局域网，为其他具有WiFi功能的设备提供互联网接入服务。机身配有1.4寸128×128像素分辨率显示屏，可显示网络信号、接入设备数量、电池寿命等信息，支持MicroSD卡拓展至最高16GB存储空间，最多支持5个设备同时连接，内置电池可提供3小时使用时间和36小时待机时间，用户可以使用微型USB接口进行充电。

新书上架



源码中国：全球IT外包新原点

作者： Cyrill Eltschinger
译者： 高博 于海东 赵勇
出版社： 机械工业出版社

软件外包已经成为中国IT产业中贡献比重最大的领域。本书通过作者对全球形势和外包行业的渊博知识、详尽记录、严密组织和精确分析，从最大的发包方所在国的视角，全面系统地阐明了为什么中国将会成为下一个全球最大的外包口岸。作者的初衷是向外国人表明中国存在的巨大机会，鼓励更多有潜力的发包企业将业务外包到中国。通过阅读本书，读者可以切换到另一种视角，从企业和国家的角度了解整个外包行业的发展情况，以及中国在这个行业中的定位和未来。这其中包括最重要的是企业应该如何改进，才能够更好地为自己的客户带来价值。

深入理解计算机系统

作者： Randal E.Bryant David O'Hallaron
译者： 龚奕利 雷迎春
出版社： 机械工业出版社

这本书从程序员的视角详细阐述计算机系统的本质概念，并展示这些概念如何实实在在地影响应用程序的正确性、性能和实用性。

书中内容的最大优点是为程序员描述计算机系统的实现细节，帮助其在大脑中构造一个层次型的计算机系统，从最底层的数据在内存中的表示到流水线指令的构成，到虚拟存储器，到编译系统，到动态加载库，到最后的用户态应用。通过掌握程序是如何映射到系统上，以及程序是如何执行的，读者能够更好地理解程序的行为为什么是这样的，以及效率低下是如何造成的。

Boost程序库完全开发指南——深入C++“准”标准库

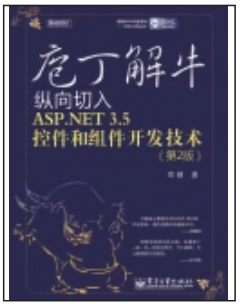
作者： 罗剑锋
出版社： 电子工业出版社

Boost是一个功能强大、构造精巧、跨平台、开源并且完全免费的C++程序库，有着“C++‘准’标准库”的美誉。

它由C++标准委员会部分成员所设立的Boost社区开发并维护，使用了许多现代C++编程技术，内容涵盖字符串处理、正则表达式、容器与数据结构、并发编程、函数式编程、泛型编程、设计模式实现等许多领域，极大地丰富了C++的功能和表现力，能够使C++软件开发更加简洁、优雅、灵活和高效。

书中内容基于Boost1.42版，介绍了其中的所有99个库，并且详细深入地讲解了其中数十个库，同时实现了若干颇具实用价值的工具类和函数，可帮助读者迅速理解掌握Boost的用法并且应用于实际的开发工作中。

庖丁解牛：纵向切入ASP.NET 3.5控件和组件开发技术



作者： 郑健
出版社： 电子工业出版社

作者主要介绍了ASP.NET的控件开发，书中通过70多个例子讲解ASP.NET控件开发技术的各个方面，剖析了很多控件中的系统基类源代码，读者从这些系统源代码可以体会设计模式思想。如果扎实地掌握了ASP.NET控件的运行机制，开发一个页面级的ASP.NET应用程序会变得非常简单。学完本书后能够掌握控件开发各个方面的技术，而且深晓ASP.NET的工作原理。对大部分使用ASP.NET技术开发两年左右的开发人员来说，本书中有75%以上的内容可能没有接触到。

作为本书的第2版，在内容方面也做了许多调整。加了第18章《基于Web的性能调优》，介绍了服务端和客户端代码性能分析、ASP.NET产品级的优化方案、网络瓶颈诊断等内容。甚至包括没有收录在书中的技巧和窍门。

计算机程序设计艺术 第1至3卷

作者：Donald E.Knuth

出版社：人民邮电出版社

《计算机程序设计艺术》系列著作对计算机领域产生了深远的影响。这一系列堪称一项浩大的工程，自1962年开始编写，计划出版7卷，目前已经出版了4卷。《美国科学家》杂志曾将这套书与爱因斯坦的《相对论》等书并列称为20世纪最重要的12本物理学著作。目前Knuth正将毕生精力投入到这部史诗性著作的撰写中。卷1为基础运算法则，该书以基本的编程概念和技术为开始，然后讲述信息结构——计算机内信息的表示法，数据元素间的结构关系以及处理它们的有效方法。第2卷讲解半数值算法，分“随机数”和“算术”两章。本卷总结了主要算法范例及这些算法的基本理论，广泛剖析了计算机程序设计与数值分析间的相互联系。第3卷扩展了第1卷中信息结构的内容，主要讲排序和查找。书中对排序和查找算法进行了详细的介绍，并对各种算法的效率做了大量的分析。

Flex 4一学就会

作者：Peter Armstrong

译者：张猛

出版社：人民邮电出版社

作为Flex 4的入门级读物，书中主要介绍了创建迷人的动画和过渡、处理用户输入、访问和显示数据、与服务器通信、使用开源组件等技巧，全书共7章。前6章由26个精心挑选、循序渐进的示例组成，主要介绍了Actionscript 3、XML、E4X、Spark组件、Spark容器、视图状态、特效、样式、Halo组件、格式化器和验证器等内容。最后讲述了如何使用Cairngorm框架并构建一个完整的Flex应用程序SocialStalkr，这个混搭程序可以在雅虎地图上显示好友的微博。另外，书中其余部分采用一个虚拟实训课的形式，趣味性强，并配有幽默漫画，使得学习过程轻松愉快。

自己动手写网络爬虫

作者：罗刚 王振东

出版社：清华大学出版社

大多数网络爬虫的开发原理与技巧在专业的公司内部都秘而不宣，至今仍然缺少理论与实践相结合的专门介绍网络爬虫的书籍。本书尝试理论与实践相结合，深入透彻地讲解网络爬虫的原理，并且辅以相关代码作为参考。本书两位主要作者在搜索引擎领域都有丰富理论和实践经验。同时，有多个程序员帮忙开发或编写了代码实现，例如Java实现异步I/O或对PDF文件的处理等。

编写本书的两位作者从基本的爬虫原理开始讲解，通过介绍优先级队列、宽度优先搜索等内容引领读者入门；之后根据当前风起云涌的云计算热潮，重点讲述了云计算的相关内容及其在爬虫中的应用，以及带偏好的爬虫、信息抽取、链接分析等内容；最后两章还介绍了有关爬虫的数据挖掘的内容。

全球排行榜

	Amazon
1	The Accidental Billionaires: The Founding of Facebook: A Tale of Sex, Money, Genius and Betrayal
2	The Facebook Effect
3	The Web Designer's Idea Book
4	Cracking the Coding Interview
5	Presentation Zen
6	Head First HTML with CSS & XHTML
7	Hello, Android: Introducing Google's Mobile Development Platform
8	iPhone Programming: The Big Nerd Ranch Guide
9	JavaScript: The Good Parts
10	Head First Java, 2nd Edition

	天珑书局（中国台湾）
1	深入浅出 Android 系统原理及开发要点
2	全球最强 VMware vSphere 4 企业环境建构
3	Google Android 2.X 应用程序开发实战
4	Google Android SDK 开发范例大全 2
5	ASP.NET MVC 2 开发实战
6	玩通 VMware：108 个虚拟机实例讲堂
7	HTML 5 & API 网页程序设计
8	约耳趣谈软件——来自项目管理的现场实录
9	嵌入式系统开发之道——菜鸟成长日志与项目经理的私房菜
10	巧用 jQuery

	第二书店
1	深入浅出数据分析
2	软件开发者路线图：从学徒到高手
3	高效人士的 116 个 IT 秘诀
4	我编程，我快乐：程序员职业规划之道
5	算法导论
6	编程之美——微软技术面试心得
7	JavaScript 高级程序设计
8	设计模式沉思录
9	并发的艺术
10	程序员修炼之道——从小工到专家

王煜全，Frost & Sullivan首席咨询顾问。

程序员的无线互联创业陷阱

从目前的情况看，无线互联的应用充满了机会，但在这些机会中隐藏了很多陷阱。作为技术开发能力强大的程序员，如何辨别这些机会和陷阱呢，这里给大家提供一些基本的判断。

一、应用开发是面向现在还是面向未来？

绝大多数程序员在开发应用的时候，只看到目前技术是什么状态、用户是什么需求，而没有更多地看未来是什么样子，这就导致在开发应用的时候，更多的是面向现在（甚至面向过去）进行开发，解决的是用户当下的需求。但是无线互联的发展将会十分迅速，新技术、新应用将层出不穷，不能面向未来进行开发，任何产品都是没有持久生命力的。

在无线互联网领域，过去以及现在的技术障碍主要有：

带宽不够。由此带来流量费过高，因此有人会针对这一点进行节约流量的应用开发，这些能够满足当前用户的需求，但是，未来的带宽一定不会是关键的问题，那么这些应用就会失去存在的理由。

不能直接访问互联网。因此很多免费WAP网站有了存在的环境，其实它起到的作用就是将互联网内容转移到无线互联上来，一旦手机能直接访问互联网，他们将毫无优势可言。

依赖不同运营商。运营商的支撑平台功能无法交叉调用，造成各个平台上的功能单调，同时运营商不向合作伙伴提供相关必要功能。因此，基于一个简单平台自己开发所有功能的应用，在未来都会受到很大的冲击。

如此说来未来是什么样子呢？WiFi手机将普及，流量将以包月套餐为主，因此大带宽应用将成为可能。脱离运营商的新无线互联产业链将会成型，大量公司会提供开放的功能接口供第三方开发者调用，解决了开发者获得基本应用能力的问题。这些都有可能是我们面向未来进行创业的机会。

二、应用开发是否挡了巨人的路？

无线互联的革命是以iPhone和Android为代表引爆的。其优秀的操作系统给了开发者足够的空间，因而很多创业者都在找Android里还没能提供的功能或者还没能做好的应用做开发，希望借着大家对Android的追捧来提升自己的价值。然而，这样做是相当危险的，因为操作系统是基本，提供操作系统的厂商必然会长期投入重兵，今天的缺陷，明天必然会被重点改进。除非创业者有打败苹果、谷歌的决心和能力，否则，就要考虑你的开发是不是挡了他们未来的路？

例如，手机浏览器是现在无线互联最火的应用之一，但Symbian已经推出了自己的浏览器，iPhone也支持自己的浏览器，我们相信Android未来也很快会自带浏览器。就像微软捆绑IE打败了Netscape一样，因为浏览器本来就是手机操作系统的一部分。因此，作为创业者，我们一定要时刻提醒自己，是不是挡了哪个“大家伙”的路？

三、应用开发是否只是局部的改造？

无线互联应用开发已经进入了“骑士”时代，这意味着两三个人就可以独立开发应用。同时也意味着任何创造力不够的应用都会被很多人想到、被很多人开发，进而在市场上出现大量相似版本的应用。这会使其中任何一个版本都难以脱颖而出。

最近，无线互联领域内出现一个新名词：“微创新”。这个概念一夜之间被广为传诵，并被很多创业者认为是找到了捷径。其实，这个概念对创业者是很大的误导，甚至会有伤害。因为“微创新”不是根本性的创新，不能形成有效的知识产权保护，很容易被更大的竞争对手“抄袭”，对弱小的创业者来说，这种抄袭是致命的。所以，对“微创新”，适用于要么市场中尚无强大的同类型竞争对手，要么本身就是大企业在做，作为初创期的创业企业或者个人开发者，光靠微创新是远远不够的。

作为个人程序开发者，找到创新机会、捕捉未来机会不是没有可能的。例如，在无线互联领域，当涌现出众多的应用时，怎么把应用系统性、针对性地推介到每个用户那里就成了商机；当SNS和微博方兴未艾的时候，更多基于人际传播和影响力传播的应用就有了生命力；当运营商的控制力逐渐减弱的时候，P2P的应用形式就有了空间……

郝培强，现居上海，程序员及二手经济学家。

未有天才和创新之前

鲁迅有过一个很有名的演讲《未有天才之前》，那是在1924年，针对一片呼唤天才的声音，他说要有天才必须有能够培育天才的土壤。我时常感觉这个跟当下的情景非常相似，我们现在有各种创新奖励基金，我们用各种姿势呼唤创新，然而创新在哪里呢？

跟1924年不同，现在中国GDP已经超过日本，名列全球第二；在次贷危机引发的全球衰退中，中国是风景这边独好。但是历史总是有惊人相似的一面，1924年人们呼唤天才，现在我们呼唤创新，这似乎说明我们和1924年的中国人一样急功近利，以为天才和创新是可以从天下掉下来的，不需要土壤、不需要培育。

创新首先需要宽松的环境。任何伟大的创新刚刚从土壤冒出来的时候，都浑身是土，看起来又脏又丑，完全不受人待见。Google公司刚创业的时候，创始人是两个刚拿到博士学位的Nerd。即使是给他们第一笔天使投资的教授，也一定不会想到Google会成长为全球最大的搜索引擎公司。和纳斯达克的大多数新兴公司一样，Google遵循了类似的路径：天使、风险投资，IPO。其中天使和风险投资部分的投资成功率非常低，经常投资100个公司只有7～8个真正成长起来，但是因为少数成功的项目的投资回报率极其可观，所以，这两种投资还是在蓬勃发展中。中国现在也有了很多天使和风险投资（有的是国外的基金，有的是本土的资金）。已经在促进创新方面起了很大的作用。但是在中国的创业投资还很保守，风险投资部分发展得还可以，但是关注于更小企业的天使投资还没有发展起来。

宽松的资金环境只是其一，更重要的是宽松的言论环境，宽松的竞争环境。Twitter是一个典型的例子，这种促进信息流动言论自由的应用绝对不会首先从中国发展起来。当Twitter成为了全世界最重要的一种互联网服务类型的时候，中国的互联网公司纷纷跟进，我们有了新浪微博，腾讯微博，网易微博，搜狐微博……等等。但如果Twitter这种形式是新浪发明的会如何？结局大家不难以想象吧？中国僵化的体制，低效的官僚体系，造成一个可笑的局面，一方面知道经济要继续发展，要提高国家的竞争力，需要创新；但另外一个方面，他们看到任何形式的创新时，第一个念头就是这东西要干什么，是不是造成麻烦。所以本质上，他们是反对创新的，虽然创新可能带来效益，甚至可能给他们带来政绩，但是在官场有功没有比无过更加重要。

中国的竞争环境也不够宽松。几乎每个初创企业被投资的时候，都会被问道，如果百度做了你的业务，或者如果腾讯做了你的业务，你该怎么办的问题。这是不良竞争环境的一个典型体现。在美国，如果你业务做得很好，证明了一个商业模式有价值以后，大公司会考虑收购，创业者和投资人会得到丰厚的回报，从而促进创新。而在中国，巨无霸企业就会考虑直接复制你的模式，创业者和投资人可能会在跟大企业的竞争中败下阵来。近两年还有一个更严重的问题，那就是国进民退。当一个国有企业倾全国之力去和中小企业竞争的时候，一切创新都会被抹杀，资本决定一切。

另一个大问题是知识产权保护方面。因为缺乏对版权的保护，中国的通用软件行业其实已经白白浪费了十多年，没有出现一家上市公司，没有一家公司真正通过卖软件获得客观的利润。同样是这十多年，所有致力于数字音乐版权、数字内容版权的公司也罕有成功的。而数字内容版权，数字音乐版权是这些年美国创新的一个热点。03年到现在，在美国苹果公司创建的数字音乐市场iTunes Music Store已经超过了线下的最大音乐销售店沃尔玛，而苹果的数字电影销售和租赁市场的销售额也在大幅增长中。亚马逊的电子书也超过了实体书的销量。而在中国，所有致力于数字音乐、数字内容的公司，还在猖獗的盗版下苟延残喘。

知识产权保护不利的另外一个方面，是专利保护的匮乏，以及对不正当竞争的漠视。创新的小公司，得不到回报，前面我们提到过“在中国，巨无霸企业就会考虑直接复制你的模式”就是因为用复制的方式成本最低，而且法律对此完全无能为力。在中国，当某些企业大到一定程度以后，就可以获得地方政府的保护，在法律上政策上得到各种倾斜，这些不公平的现象，都大大地放大了马太效应，导致强者更强，有创新的小公司很难脱颖而出。

高巍，安卓爱普公司创始人。原搜狐媒体产品中心经理。关注移动互联网、互联网产品管理。

创业大败局

——25个创业者失败案例的启示

Y-Combinator创始人、著名天使投资人Paul Graham投资了80多家创业公司、堪称互联网创业领域的教父级人物，经常被人问到“哪些错误会导致创业失败”。Paul Graham尝试开出了一张“创业失败的18个错误”的不完全清单，希望帮助创业者察觉到自己正在做不应该做的事情。最近，一篇文章《25则“验尸报告”——创业失败者启示录》（http://news.csdn.net/a/20101018/280604.html）也受到关注，作者的出发点与Paul Graham类似，虽然成功不可以复制，失败却可以尽量避免。

有趣的是，文章中提到的创业者们并不同意这个看法。YouCastr创始人说他读过Paul Graham的几乎每一篇文章，但不亲身经历一遍是很难体会其中的道理。他创业失败后在的经验总结中写道，希望自己的特定经历与个人体验能成为Paul Graham创业文章的一个注脚、一个背景诠释。IonLab创始人则更是夸张，建议创业者们停止阅读一切创业指导文章或图书，因为每一个小时的阅读，至少需要创业者此前有三个小时以上的实际经验才能理解其中内容，又至少再需要三个小时的实践运用才能吸收。即使这样，也很难把这些经验教训“内化（internalize）”。IonLab创始人说，他读Paul Graham文章时一直想，这些不会发生在自己身上，轮到自己创业一定能做得更好。当他创业失败，重读Paul Graham时，却发现说的就是自己，惊出一身冷汗。

从某种意义上说，导致创业失败的本质错误只有一个——没人需要你做的东西。如果你在做的东西是用户需要的，那么你至少能够生存下去，其它的问题都可以逐一解决。但如果不符合用户需求，那就死定了。

25则创业者失败教训总结，几乎都有一个共同点：个人有创业冲动，正好想到了一个自以为绝妙的点子，或看到了一个自认为很大的机会，就不管不顾地投入进去，而没有进行客户需求的确认。

eHarmony for jobs创始人想到了一个自动匹配求职者与用人企业的点子，仅仅咨询了身边朋友，得到认可后就迫不及待地投入创业。在这个过程中才认识到，“他们（指自己的朋友）只代表了不需要付费的求职者，而最重要的需要付费的企业客户却没有去调研”。甚至在深入HR行业一段时间后，才知道这是一个过去几年被无数人想到、尝试并验证失败的模式。

BricaBox创始人最终认识到自己要解决的只是一个“技术问题”，而不是“商业问题”。技术人员创业，“自挠其背而止痒”当然没错，但最好是有“一个更具通用性的可推而广之的解决方案”。他总结道，解决技术问题，更应该是发起一个开源项目，而不是创办一家公司。

Xmarks代表着典型的技术人员创业。Mozilla基金会主席，自己却用Safari并非Firefox，因为需要在5台不同电脑上工作，而当时Firefox没有书签同步功能，就开发了Xmarks。他们当时的想法是，用户收藏网页相当于一次投票，可以做出一个更好、更智能的搜索引擎。请了可用性专家和用户测试，才发现“人们搜索是想找到特定问题的答案，而不是得到某个主题领域内的一组权威链接”，并“惊讶”地发现“做搜索测试时，人们在电脑前坐下来第一件事就是搜索自己的姓名”。

YouCastr创始人的总结更是沉痛，“我们并不爱自己做的事。我们做这个只是因为我们想创业，想到了这个点子觉得不错。我们不是我们产品的核心用户。虽然我们很努力工作，但我们无法理解什么是最佳的产品决策。”“创业基本前提从一开始就错了，我们既没有调查视频内容提供方，也不了解观众。”而认识到这点，创业团队用了三年时间，其中整整两年，5个创始人都没有领工资。

最近流行的“Lean Startup（精益创业）”，其精髓是“Customer Validation（客户需求确认）”，即用最低成本、尽早获取客户需求的确认。推荐的方法都很有意思，如做一个假的网页，投放Adwords关键词广告，通过点击数据分析来判断用户是否真的需要你准备投入创业的这个产品；又如可以先不做产品，制作一段演示视频，扔到网上传播看用户如何反馈。

创业大败局给我们的启示，借用Paul Graham的话，那就是“理解你的用户”。